(2)

NAVAL POSTGRADUATE SCHOOL
Monterey, California

THESIS

A COMPUTER ANALYSIS OF PROPORTIONAL NAVIGATION AND COMMAND TO LINE OF SIGHT OF A COMMAND GUIDED MISSILE FOR A POINT DEFENCE SYSTEM

by

Dimitrios Ioannis Peppas

December, 1992

Thesis Advisor: Harold A. Titus

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1a REPORT SECURITY CLASSIFICATION UNCLASSIFIED	1b RESTRICTIVE MARKINGS
2a SECURITY CLASSIFICATION AUTHORITY	3 DISTRIBUTION AVAILABILITY OF REPORT Approved for Public Release; distribution is unlimited.
2b DECLASSIFICATION DOWNGRADING SCHEDULE	
4 PERFORMING ORGANIZATION REPORT NUMBER(S)	5 MONITORING ORGANIZATION REPORT NUMBER(S)
6a NAME OF PERFORMING ORGANIZATION Naval Postgraduate School 6b OFFICE SYMBOL (If applicable) EC	7a NAME OF MONITORING ORGANIZATION Naval Postgraduate School
6c ADDRESS (City, State, and ZIP Code) Monterey, CA 93943	7b ADDRESS (City, State, and ZIP Code) Monterey, CA 93940
8a NAME OF FUNDING/SPONSORING ORGANIZATION 8b OFFICE SYMBOL (If applicable)	9 PROCUREMENT INSTRUMENT IDENTIFICATION NUMBER
8c ADDRESS (City, State, and ZIP Code)	10 SOURCE OF FUNDING NUMBERS PROGRAM ELEMENT NO / PROJECT NO / TASK NO / WORK UNIT ACCESSION NO

11 TITLE (Include Security Classification) A Computer Analysis of Proportional Navigation and Command to Line of Sight of a Command Guide ED Missile for A Point Defence System.

12 PERSONAL AUTHOR(S) Dimtrios Ioannis Peppas

13a TYPE OF REPORT	13b TIME COVERED	14 DATE OF REPORT (Year, Month Day)	15 PAGE COUNT
Master's THesis	FROM _____ TO _____	December 1992	137

16 SUPPLEMENTARY NOTATION The views expressed in this thesis are those of the author and do not reflect the official policy or position of the Department of Defense or the United States Government.

17 COSATI CODES			18 SUBJECT TERMS (Continue on reverse if necessary and identify by block number)
FIELD	GROUP	SUB-GROUP	Proportional Navigation Command Guidance, Command to Line of Sight Guidance, Missile Guidance, Missile Simulation

19 ABSTRACT (Continue on reverse if necessary and identify by block number)

This thesis compares two types of command guidance to be used by a point defense system: Proportional Navigation and Command to Line Of Sight (CLOS). The system block diagram was first defined. The necessary transfer functions were derived. Two forward time models were evaluated, one for each guidance method, using state variable analysis. Two three dimensional scenarios were defined and their results used to compare the two methods. Parameters considered in the comparison were miss distance and acceleration load on the missile.

20 DISTRIBUTION AVAILABILITY OF ABSTRACT ☒ UNCLASSIFIED UNLIMITED ☐ SAME AS RPT ☐ DTIC USERS	21 ABSTRACT SECURITY CLASSIFICATION UNCLASSIFIED
22a NAME OF RESPONSIBLE INDIVIDUAL Titus, Harold A.	22b TELEPHONE (Include Area Code) 408-646-2560 22c OFFICE SYMBOL EC/Ts

DD Form 1473, JUN 86 Previous editions are obsolete

Approved for public release; distribution is unlimited.

A Computer Analysis of Proportional Navigation
and Command to Line of Sight of a Command
Guided Missile for a Point Defence System

by

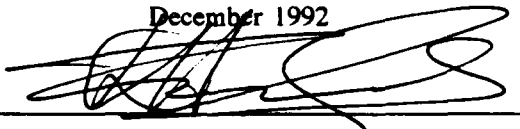
Dimitrios I. Peppas
Lieutenant (junior grade), Hellenic Navy
B.S.E.E., Hellenic Naval Academy, 1984

Submitted in partial fulfillment
of the requirements for the degree of

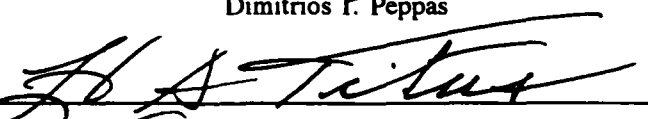
MASTER OF SCIENCE IN ELECTRICAL ENGINEERING

from the

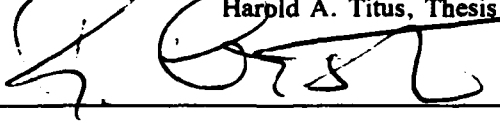
NAVAL POSTGRADUATE SCHOOL
December 1992

Author: 

Dimitrios I. Peppas

Approved by: 

Harold A. Titus, Thesis Advisor



Roberto Cristi, Second Reader

Michael A. Morgan, Chairman
Department of Electrical and Computer Engineering

ABSTRACT

This thesis compares two types of command guidance to be used by a point defence system: Proportional Navigation and Command to Line Of Sight (CLOS). The system block diagram was first defined. The necessary transfer functions were derived. Two forward time models were evaluated, one for each guidance method, using state variable analysis. Two three dimensional scenarios were defined and their results used to compare the two methods. Parameters considered in the comparison were miss distance and acceleration load on the missile.

TABLE OF CONTENTS

ACKNOWLEDGMENTS

I would like to thank God, for showing me the way, my father, who has always been there for me, and my wife Lena and my daughter Terry, for reminding me what it's all about.

I have had the privilege and the honor to have had Professor Hal Titus as my teacher and as an advisor. His knowledge and devotion to his work is only surpassed by his dedication to his students.

I would also like to thank the Hellenic Navy, who gave me the chance to fulfill a lifetime dream.

Finally to the organization known as the Naval Postgraduate School, for giving me not only an education, but most of all, a new way of thinking.

I. INTRODUCTION

The term "guidance" implies that the missile responds to steering commands in order to improve it's accuracy in delivering the warhead. These commands issued to the missile can either be internal or external. When the commands are internal the missile incorporates a seeker that tracks the target and a guidance system that translates the target data to steering commands. This being the case, the missile is independent from the platform it was fired from. The name associated with this type of missile is "Fire and Forget". When the commands are external the target data can be collected by a Tracker/Fire Control System on the firing platform. The guidance system calculates the desired steering commands. These are transmitted to the missile. This implies that the missile is dependent on the firing platform throughout the encounter. Command is the name of this type of guidance. A generic system is shown in Figure 1 [Ref. 1: pp.27].

There are several types of guidance algorithms: proportional navigation, beam rider or command-to-LOS, etc.

The missile incorporates an autopilot by which the signals are transformed, via actuators, into turning moments.

The missile is assumed to maneuver via "skid-to-turn".

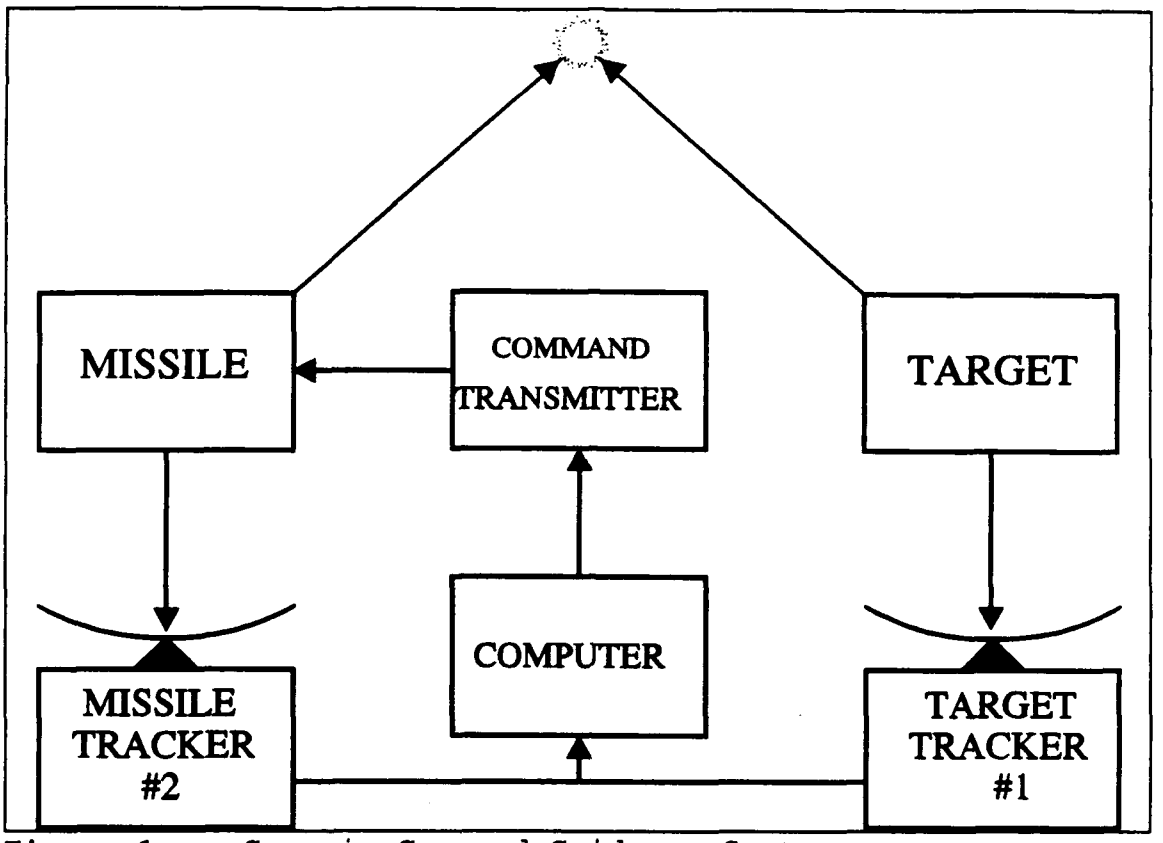


Figure 1. Generic Command Guidance System

This is the case for tactical missiles. When the target is at long range, the missile may maneuver by a "bank-to-turn" maneuvering law. This is a "minimum control effort" concept that increases the effectiveness of the fuel consumption and thus range.

This work develops a missile/target simulation program using proportional navigation and beam rider command guidance. A three dimensional model is constructed for a more realistic approach to the problem. Chapter II introduces the command guidance theory. Chapter III develops the transfer functions of the system, the problem geometry, and the

relationships that are to be simulated. Chapter IV shows the development of the computer code and relays the simulation results. Chapter V discusses the conclusions and recommendations.

This simulation uses MATLAB. The three dimensional plots are generated from GRAFTOOL.

II. COMMAND GUIDANCE

A. GENERAL

Our initial approach will be two dimensional [Ref. 2]. Consider the geometry in Figure 2. The tracking radar (usually referred to as the illuminator) measures the following values:

R_m : The illuminator-to-missile range.

R_t : The illuminator-to-target range.

σ_m : The illuminator-to-missile line of sight angle.

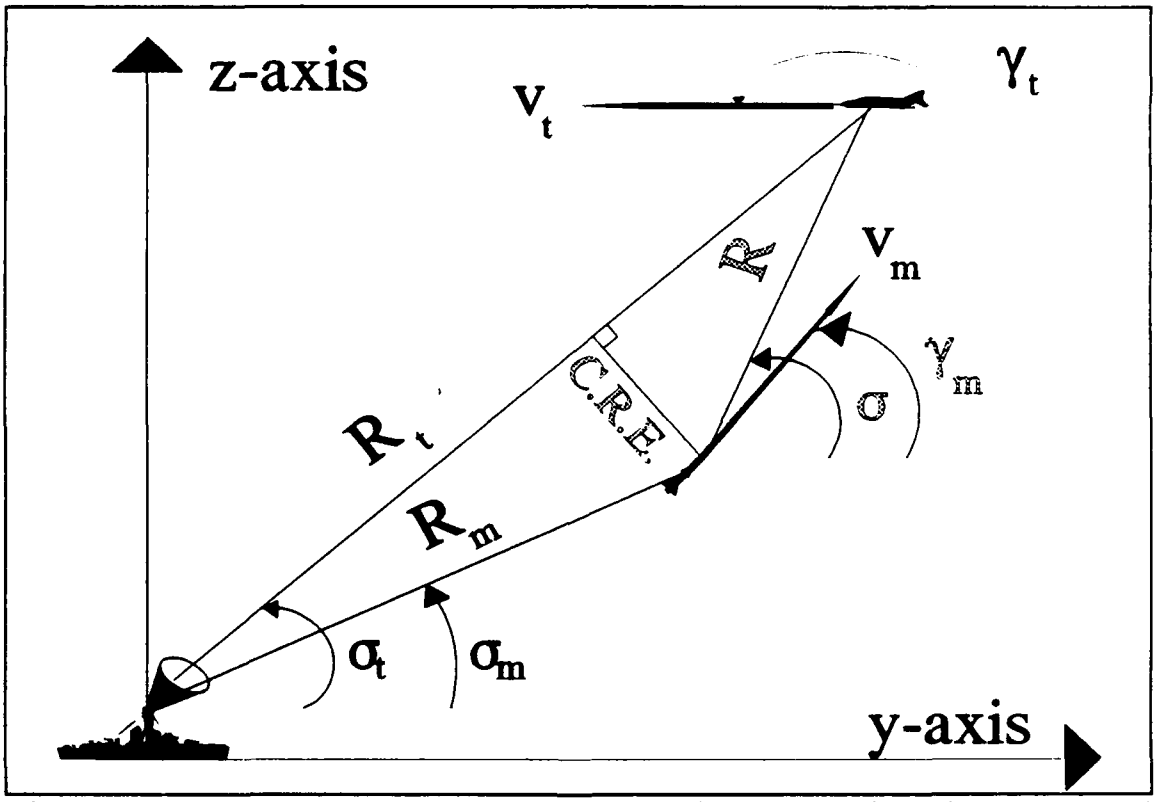


Figure 2. Fundamentals of Proportional Navigation Command and Command-to-Line-Of-Sight Guidance

σ_t : The illuminator-to-target line of sight angle.

CRE : The missile Cross Range Error from the target tracking beam.

From these measurements, the Fire Control System solves for the following values :

R : The missile-to-target range.

R' : Rate of change of R.

σ : The missile-to-target line of sight angle.

σ' : Rate of change of σ.

The guidance system of the Fire Control System now requires that an interception can occur. This can happen by driving:

$$\begin{matrix} \dot{\sigma} = 0 \\ \dot{R} < 0 \end{matrix} \qquad (2.1)$$

or, by driving:

$$\begin{matrix} CRE = 0 \\ \dot{R} < 0 \end{matrix} \qquad (2.2)$$

More insight on these relationships in the next section. But, for now, the commands issued to the missile will affect the following values:

v_m : The missile velocity.

γ_m : the missile flight path angle.

Finally, we know that the solution to the problem also depends on:

v_t : The target velocity.

γ_t : The target flight path angle.

We will use here the Line of Sight (LOS) as that between the target and the missile. This so as to reduce the quantity of subscripts.

B. INTERCEPTION THEORY

The problem is to find a way by which the missile can hit the target. Figure 3 shows several techniques developed to solve this [Ref. 3: pp.349]. " Tail Chase", also known as "Pure Pursuit", is the trajectory run through by a missile aiming at the instantaneous position of the target. "Three

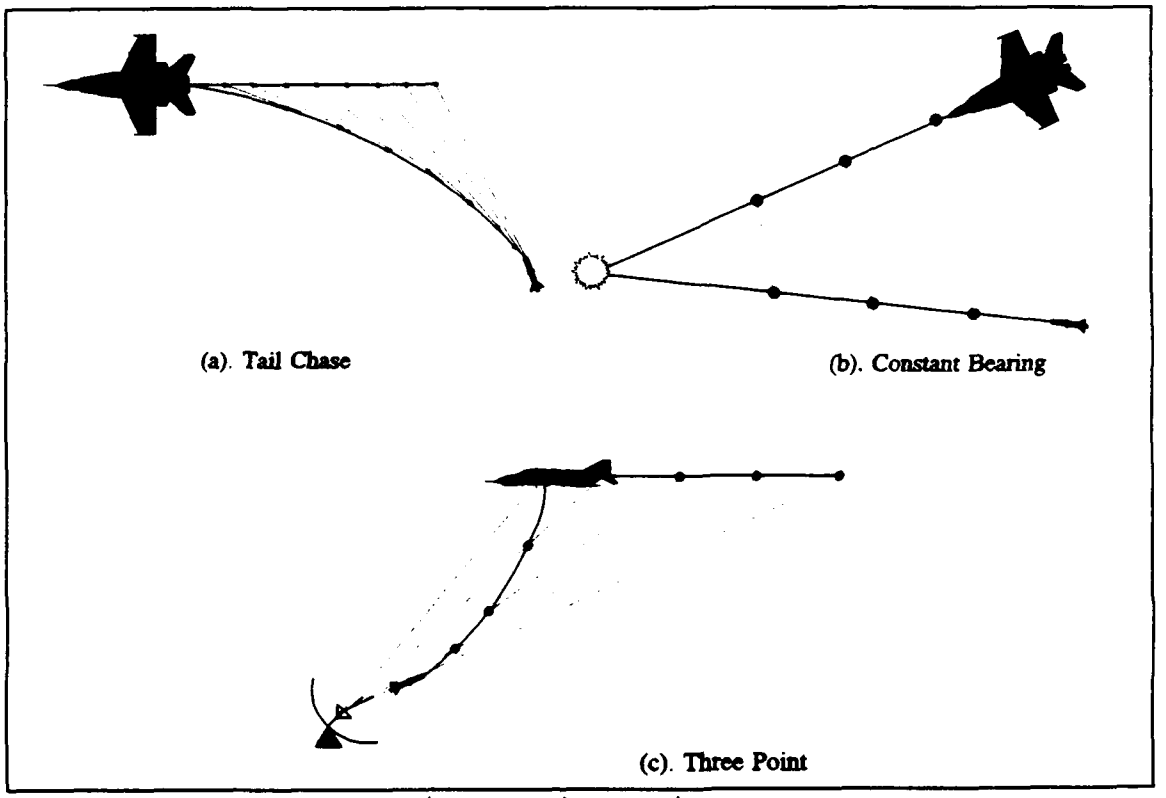


Figure 3. Interception Trajectories

Point" is the trajectory run through by the missile requiring the three points: illuminator, missile and target to be on the same line (LOS) at all times. These two techniques require a highly maneuverable missile, able to withstand high accelerations, induced by the large change of the missile velocity vector, during the final seconds before impact (this period is usually referred to as the "endgame").

Three point guidance is usually referred to as Command-to-Line-Of-Sight (CLOS). This is based on the minimization of the Cross Range Error (CRE);i.e. the displacement of the missile from the illuminator-to-missile line of sight. The theory of this technique is that since the three points are always on the same line and the missile-to-target distance is always shortening, these two are bound to meet. This explains the relationships of Equations 2.2 of the previous section.

The simplest technique of all is the "Constant Bearing", also known as "Optimum Pursuit". The theory behind this technique is simple; if the missile "sees" the target at a constant bearing ($\sigma'=0$) and if the distance between the two is continuously closing ($R'<0$), then the missile and target are bound to collide. This explains the relationships of Equations 2.1 of the previous section. We define the closing velocity as the negative rate of the LOS range:

$$v_c = -\dot{R} \qquad (2.3)$$

The benefits of this technique lie in the smaller acceleration requirements during the endgame phase. Less missile energy is also required [Ref. 4: pp. 26]. The guidance law associated with this technique is called proportional navigation.

We will first look at the importance of the LOS angle rate, since this is a major factor in this law. Miss distance is the minimum missile-to-target distance;i.e. it is the range at the Closest Point of Approach (CPA). Figure 4 displays the configuration examining the miss distance [Ref. 2: pp.33-34]. It can be defined as:

$$R_{miss} = \min[R(t)] \qquad (2.4)$$

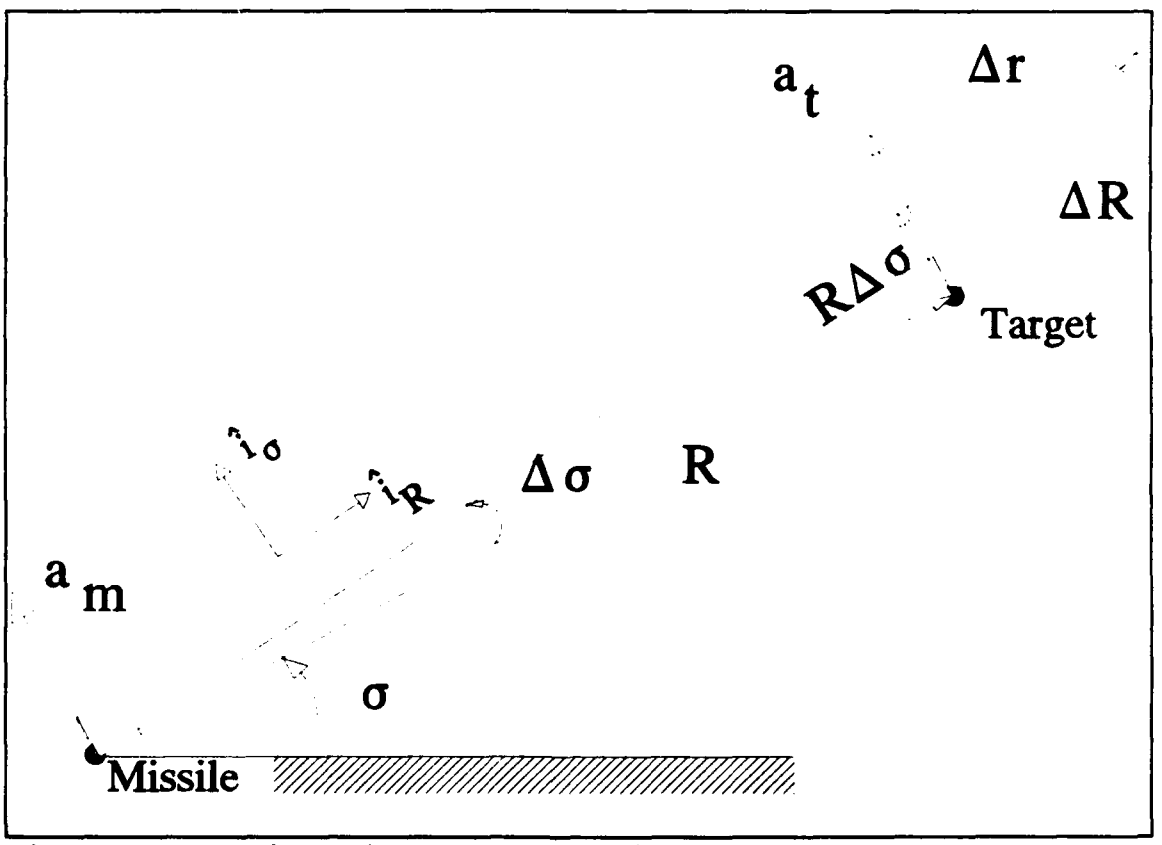


Figure 4. Miss Distance Analysis

From vector theory, a vector quantity $\underline{A}$, rotating with an angular rate $(\underline{\omega}=d\underline{\sigma}/dt)$, is differentiated by the following law:

$$\frac{d\underline{A}}{dt} = \frac{\partial \underline{A}}{\partial t} + \underline{\omega} \times \underline{R} \qquad (2.5)$$

We must note here that $\partial \underline{A}/\partial t$ is a vector parallel to $\underline{A}$ and that $\underline{\omega} \times \underline{R}$ is perpendicular. This leads to the incorporation of the orthogonal components $\hat{i}_R$ and $\hat{i}_\sigma$, as can be seen in Figure 4. Also from the same Figure 4, we can see that, an incremental increase in the LOS angle $(\Delta\sigma)$, modifies the LOS range rate:

$$\begin{aligned} \frac{\Delta \underline{R}}{\Delta t} &= \frac{\Delta r}{\Delta t}\hat{i}_R + R\frac{\Delta \sigma}{\Delta t}\hat{i}_\sigma \\ &= \frac{\Delta \underline{r}}{\Delta t} + R\frac{\Delta \underline{\sigma}}{\Delta t} \end{aligned} \qquad (2.6)$$

Taking the limit, of the above, as $t \to 0$, we have the derivative of the LOS range. Following the law of Equation 2.5 we get:

$$\dot{\underline{R}} = \dot{\underline{r}} + \underline{\omega} \times \underline{R} \qquad (2.7)$$

Finding the acceleration of this rate again requires the use of Equation 2.5, thus:

$$\frac{d\dot{\underline{R}}}{dt} = \frac{\partial}{\partial t}(\dot{\underline{r}} + \dot{\underline{\alpha}} \times \underline{R}) + \dot{\underline{\alpha}}(\dot{\underline{r}} + \dot{\underline{\alpha}} \times \underline{R}) \qquad (2.8)$$

$$\ddot{\underline{R}} = \ddot{\underline{r}} + 2\dot{\underline{\alpha}} \times \dot{\underline{r}} + \ddot{\underline{\alpha}} \times \underline{R} + \dot{\underline{\alpha}} \times \dot{\underline{\alpha}} \times \underline{R}$$

This acceleration is the vectorial difference of the missile and target accelerations;i.e.:

$$\underline{a}_{target} - \underline{a}_{missile} = \ddot{\underline{R}} \qquad (2.9)$$

Componentwise, in the cross-range direction (parallel to $\hat{i}_\alpha$):

$$\underline{a}_{t_\alpha} - \underline{a}_{m_\alpha} = (R\ddot{\alpha} + 2\dot{R}\dot{\alpha})\hat{i}_\alpha \qquad (2.10)$$

and in the range direction (parallel to $\hat{i}_R$):

$$\underline{a}_{t_R} - \underline{a}_{m_R} = (\ddot{R} - R\dot{\alpha}^2)\hat{i}_R \qquad (2.11)$$

In order to have a successful intercept, the cross range rate of the LOS must be zero, thus from Equation 2.7, the quantity $\dot{\underline{\alpha}} \times \underline{R}$ must be zero, implying either the LOS range or the LOS angle rate must be zero. The case of them being parallel is not physically attainable. Since the range is generally not equal to zero, we are left with the zero LOS angle rate. This implies that the LOS angle is constant and also proves the first part of Equation 2.1. Furthermore, Equations 2.10-11 show that constant LOS angle implies equal missile and target normal acceleration components. Radial acceleration components difference gives the closing

acceleration. We are now ready to develop the proportional navigation law.

C. PROPORTIONAL NAVIGATION LAW

The acceleration command (a_c) issued to the missile, by the proportional navigation law, is perpendicular to the LOS. The actual missile acceleration (a_m) is perpendicular to the missile's velocity (v_m). Figure 5 depicts these relationships [Ref. 11: pp. 8-11]. We now seek a connection between the missile's acceleration and its flight path angle. Figure 6 depicts this relationship. Given the missile velocity at a

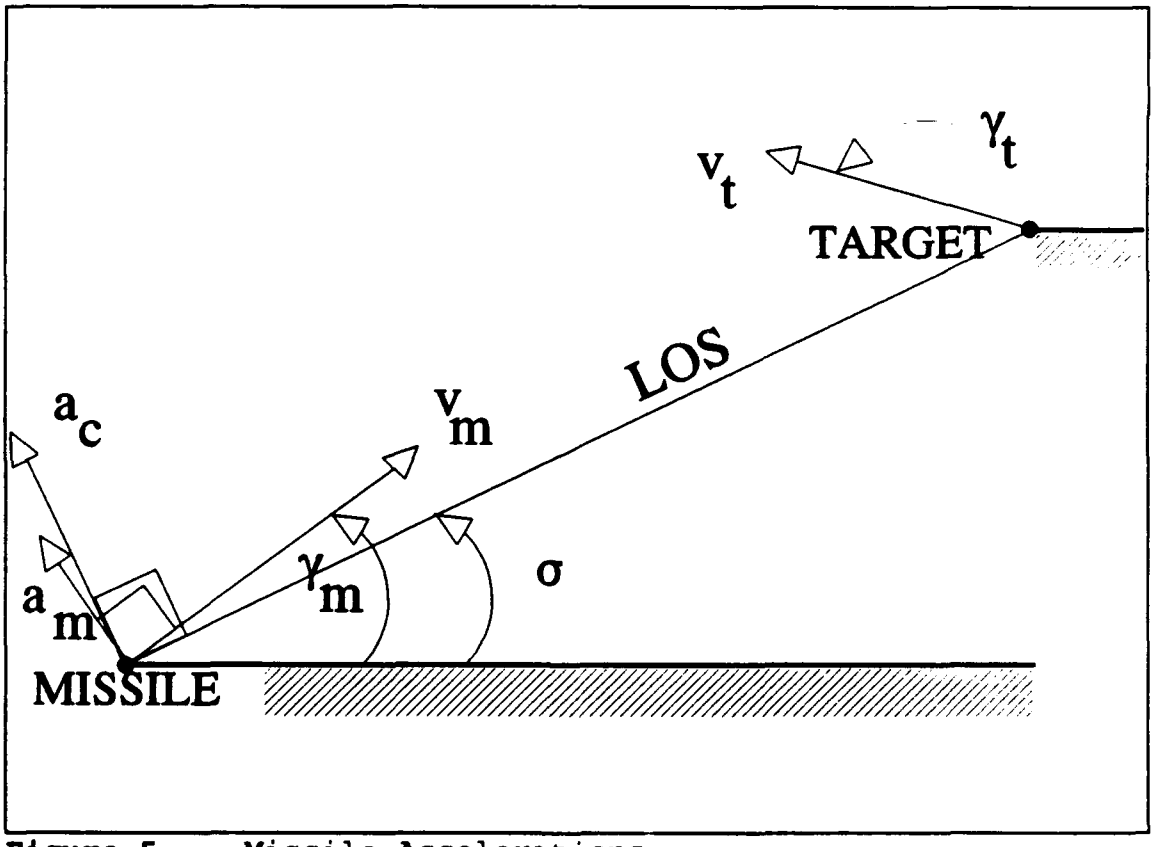


Figure 5. Missile Accelerations

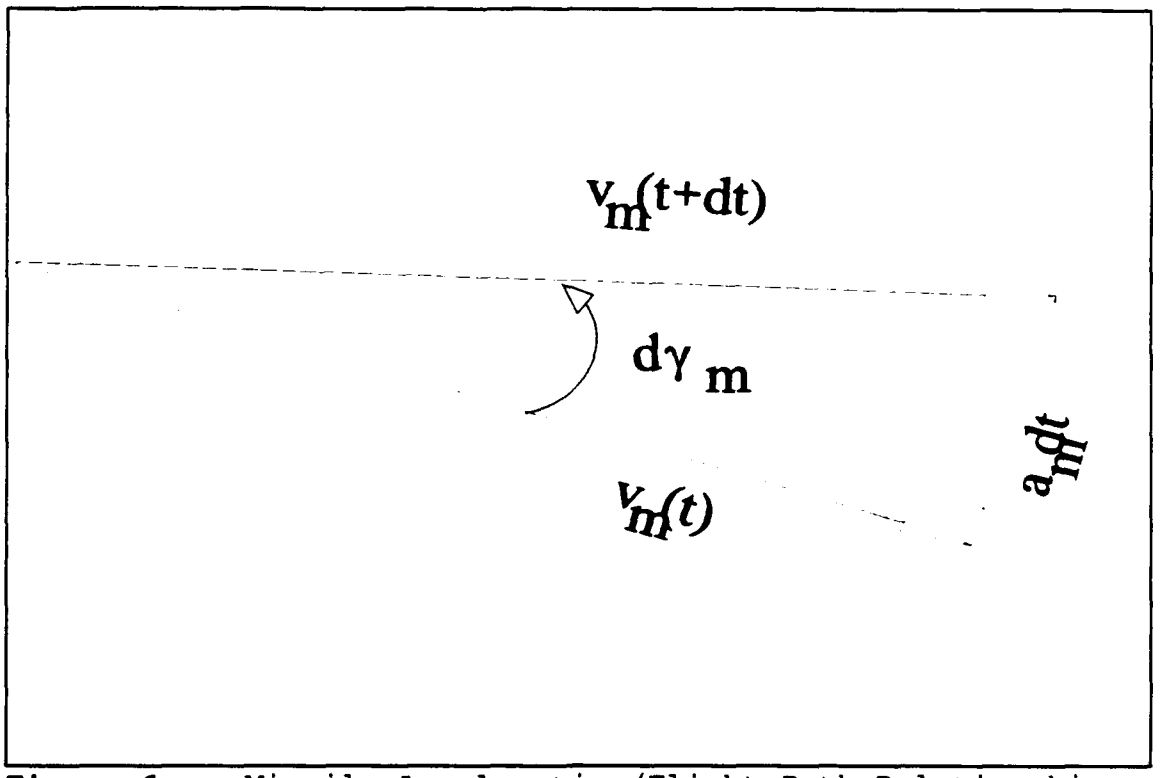


Figure 6. Missile Acceleration/Flight Path Relationship

time $t(v_m(t))$, assume an acceleration (a_m) applied on the missile for a period of dt. The velocity at the end of this period is thus $v_m(t+dt)$. The flight path angle has been changed by $d\gamma_m$. Assuming a small angle approximation, a relationship can be stated as follows:

$$a_m dt = v_m d\gamma_m \quad \Rightarrow$$
$$a_m = v_m \frac{d\gamma_m}{dt} \qquad (2.12)$$

But the proportional navigation law is given by:

$$\dot{\gamma}_m = N\dot{\sigma} \qquad (2.13)$$

where N is the proportional navigation constant. The law

states that the rate of change of the missile's flight path angle is proportional to the rate of change of the LOS angle. Substituting Equation 2.12 into Equation 2.13 yields:

$$a_m = v_m N \dot{\sigma} \qquad \textbf{(2.14)}$$

The relationship connects the missile acceleration to the LOS angular rate. The navigation constant usually ranges between 2 and 6.

D. CLOS LAW

The object of beam riding is to fly the missile along the beam that is continuously pointing at the target. The command to the missile is again an acceleration. From Figure 2 we have:

$$CRE = R_m \sin(\sigma_t - \sigma_m) \qquad \textbf{(2.15)}$$

Thus the simplest implementation of a guidance law for a beam rider system the missile acceleration command (a_{cmd}) must be proportional to the cross range error, thus:

$$a_{cmd} = K \cdot CRE = K \cdot R_m \sin(\sigma_t - \sigma_m) \qquad \textbf{(2.16)}$$

where K is the guidance gain [Ref. 2: pp. 45].

III. SYSTEM DEVELOPMENT

A. OVERVIEW

In order to approach the simulation problem we will first develop a block diagram of the system. The transfer functions of each block will then be defined. The problem geometry for the specific application will then be developed. Finally the formulas that will be used in the simulation will be given.

The generic block diagram of a command guidance system was given in Figure 1. A more specific block diagram is shown in Figure 7.

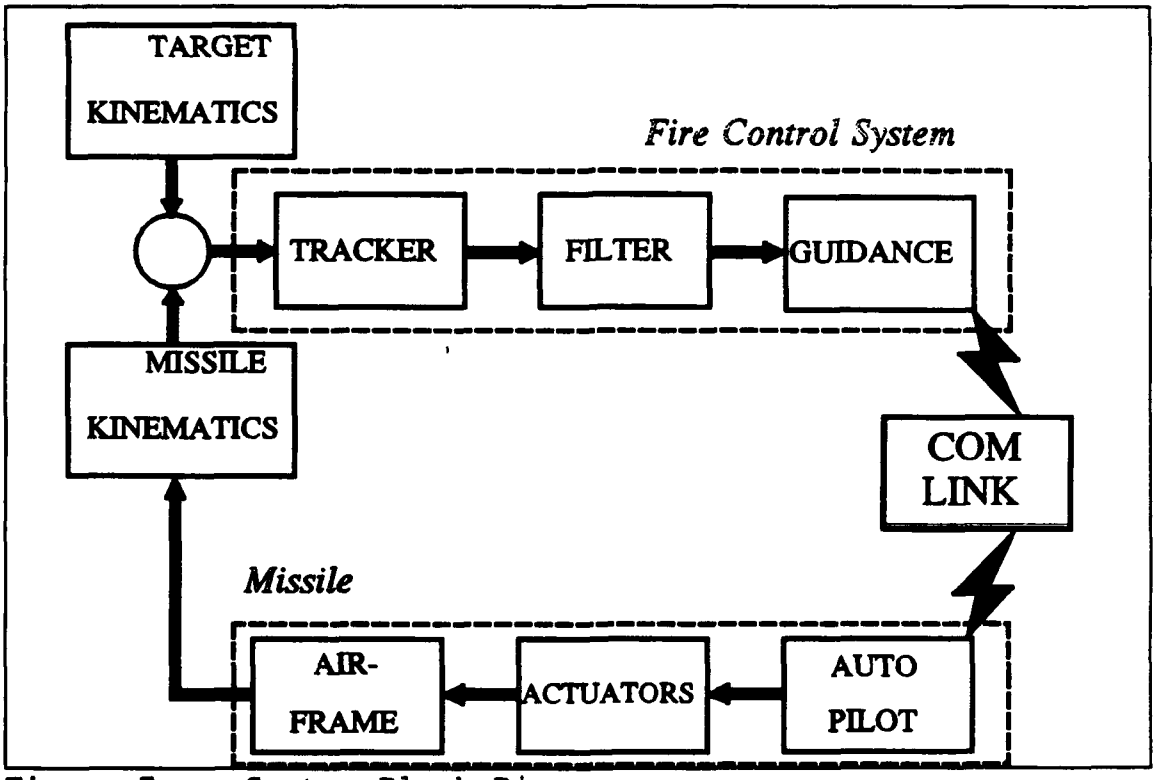


Figure 7. System Block Diagram

B. TRACKER DEVELOPMENT

A tracker receives the radar return from the target, and produces the illuminator-to-target range, and yaw and pitch angles.

The yaw plane is defined as the xy-plane. The pitch plane is the vertical plane containing the target and the missile.

1. Proportional Navigation

The tracker geometry is shown in Figure 8. From this it can be seen that:

σ_{my} : The missile yaw angle

σ_{ty} : The target yaw angle

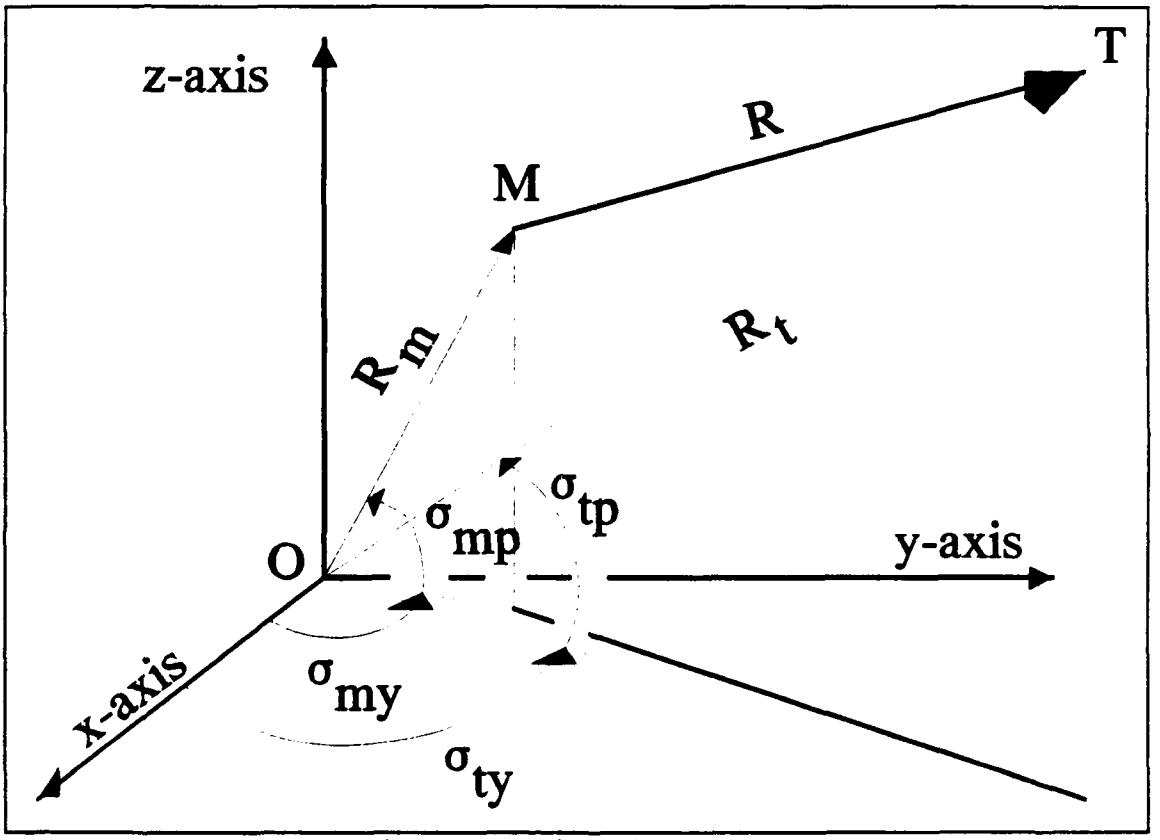


Figure 8. Proportional Navigation Tracker Geometry

σ_{mp} : The missile pitch angle

σ_{tp} : The target pitch angle

R_m : The illuminator-to-missile range

R_t : The illuminator-to-target range

R : The missile-to-target range, also known as the Miss Distance

If the missile is defined by the triplet $M(x_m, y_m, z_m)$ and the target by the triplet $T(x_t, y_t, z_t)$ then, the tracker calculates the following relations:

$$\begin{aligned}
\sigma_{m_{yaw}} &= \tan^{-1}\left(\frac{y_m}{x_m}\right) \\
\sigma_{t_{yaw}} &= \tan^{-1}\left(\frac{y_t}{x_t}\right) \\
\sigma_{m_{pitch}} &= \tan^{-1}\left(\frac{z_m}{\sqrt{x_m^2 + y_m^2}}\right) \\
\sigma_{t_{pitch}} &= \tan^{-1}\left(\frac{z_t}{\sqrt{x_t^2 + y_t^2}}\right) \\
R_m &= \sqrt{x_m^2 + y_m^2 + z_m^2} \\
R_t &= \sqrt{x_t^2 + y_t^2 + z_t^2}
\end{aligned} \qquad (3.1)$$

These are the simulation equations. Inversely, if the tracker values are known, the triplets can be calculated as follows:

$$
\begin{aligned}
x_m &= (R_m \cos\sigma_{m_{pitch}})\cos\sigma_{m_{yaw}} \\
y_m &= (R_m \cos\sigma_{m_{pitch}})\sin\sigma_{m_{yaw}} \\
z_m &= R_m \sin\sigma_{m_{pitch}} \\
x_t &= (R_t \cos\sigma_{t_{pitch}})\cos\sigma_{t_{yaw}} \\
y_t &= (R_t \cos\sigma_{t_{pitch}})\sin\sigma_{t_{yaw}} \\
z_t &= R_t \sin\sigma_{t_{pitch}}
\end{aligned}
\qquad (3.2)
$$

The tracker must produce the following values:

σ_{yaw} :The missile-to-target yaw angle

σ_{pitch} :The missile-to-target pitch angle

R :The Miss Distance

These values are shown in Figure 9. The above required quantities are incorporated in the solution frame. This is defined as the moving frame, always parallel to the reference frame, but originating on the missile. The additional equations required to be solved are:

$$
\begin{aligned}
\sigma_{yaw} &= \tan^{-1}\left(\frac{y_t - y_m}{x_t - x_m}\right) \\
\sigma_{pitch} &= \tan^{-1}\left(\frac{z_t - z_m}{\sqrt{(x_t - x_m)^2 + (y_t - y_m)^2}}\right) \\
R &= \sqrt{(x_t - x_m)^2 + (y_t - y_m)^2 + (z_t - z_m)^2}
\end{aligned}
\qquad (3.3)
$$

The combined set of Equations 3.3 are the output of the tracker for a proportional navigation fire control system, solving the three-dimensional interception.

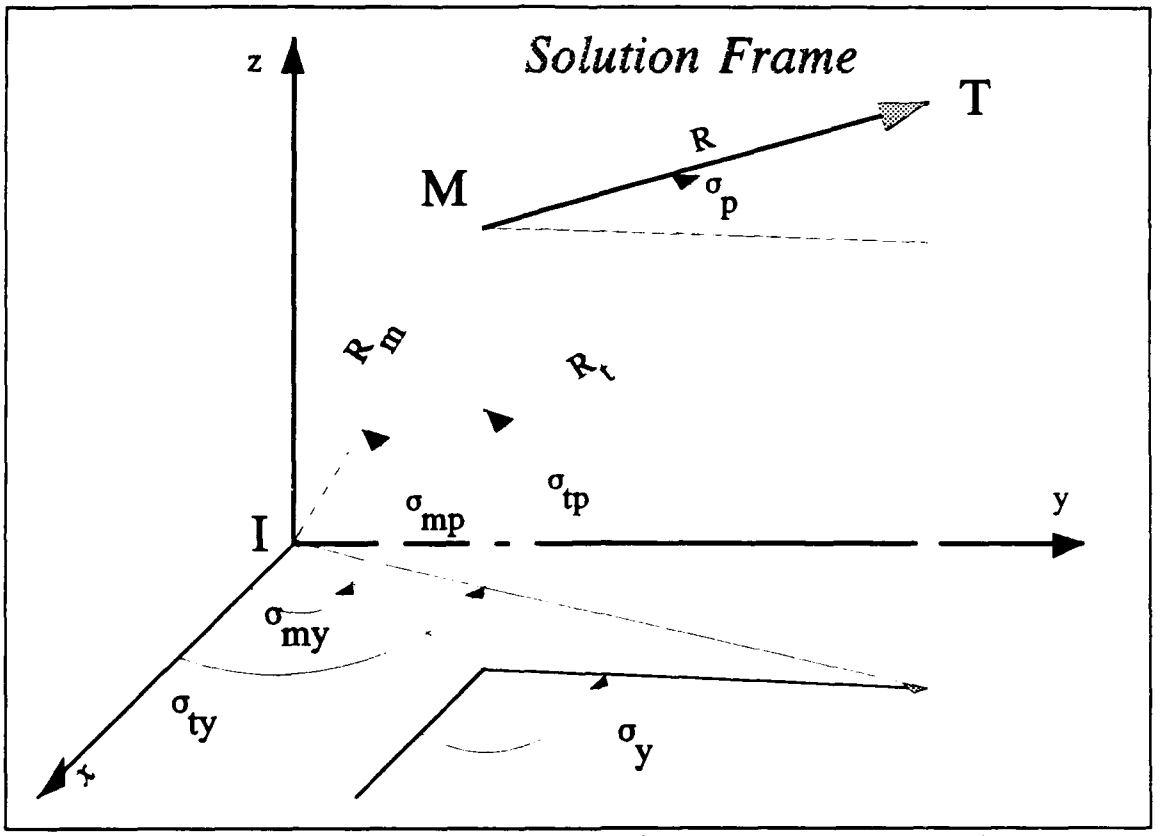


Figure 9. Proportional Navigation Tracker Solution Frame Geometry

2. CLOS

In this scheme, the tracker is required to produce the cross range error. The tracker geometry is presented in Figure 10.

From analytic geometry, the distance of a point from a line, in 3D space, is given by [Ref. 6]:

$$|CRE| \doteq CRE = \frac{|R_m \times R_t|}{|R_t|} \qquad (3.4)$$

In this case, the point is the missile and the line is the illuminator-to-target range. Substituting the triplets for the

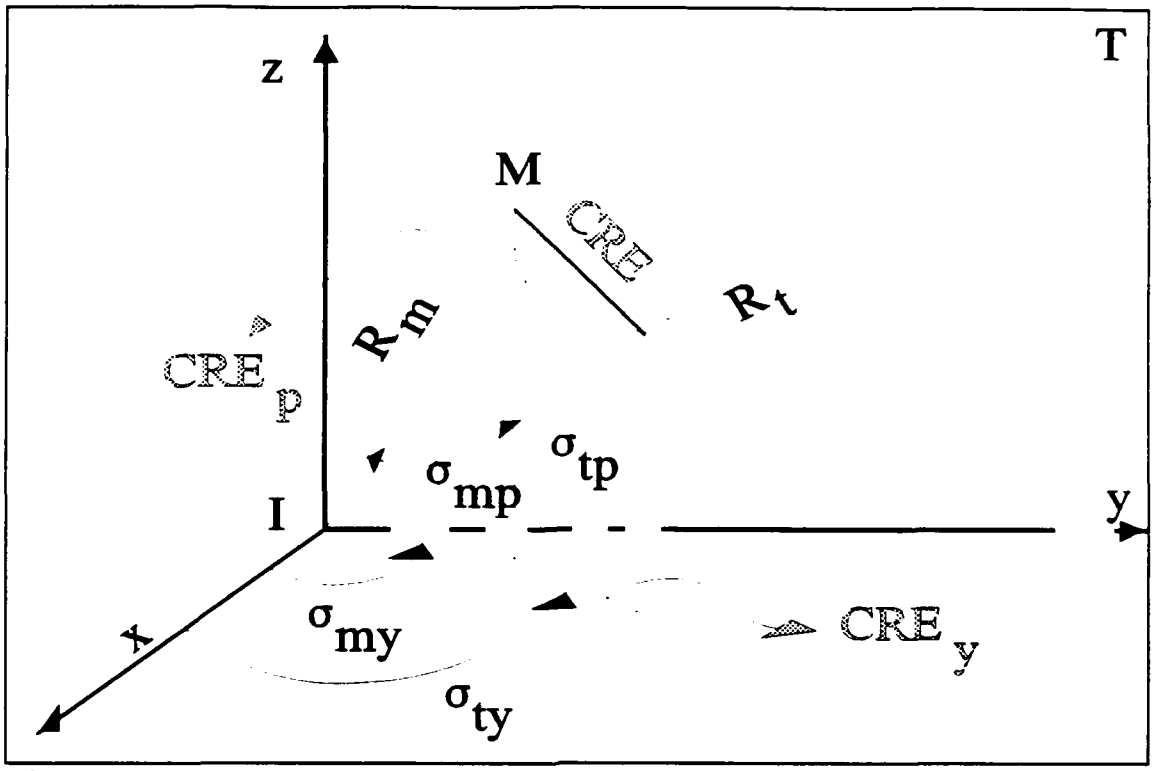


Figure 10. CLOS Tracker Geometry

missile, $M(x_m, y_m, z_m)$, and the target, $T(x_t, y_t, z_t)$, the following closed form solution for the cross range error, is derived:

$$CRE = \frac{1}{R_t}\sqrt{(x_m y_t - x_t y_m)^2 + (y_m z_t - y_t z_m)^2 + (z_m x_t - z_t x_m)^2} \quad (3.5)$$

The CRE components in the yaw and pitch plane can also be calculated. Referring to Figure 10:

$$\begin{aligned} CRE_{yaw} &= \sqrt{x_m^2 + y_m^2}\,\sin(\sigma_{t_{yaw}} - \sigma_{m_{yaw}}) \\ CRE_{pitch} &= \sqrt{CRE^2 - CRE_{yaw}^2}\,sign(\sigma_{t_{pitch}} - \sigma_{m_{pitch}}) \end{aligned} \quad (3.6)$$

The signum function is introduced in order to have a sign for the pitch cross range error. This sign is transferred to the commanded acceleration.

The combined set of Equations 3.5 and 3.6 are the output of the tracker for a command-to-line-of-sight fire control system, solving the three-dimensional interception.

C. FILTER DEVELOPMENT

1. Proportional Navigation

Proportional navigation guidance requires the rate of change of the LOS angle. The filter estimates this rate from the LOS angle observed by the missile.

The torque that will be applied to the missile will be analogous to this estimated angular acceleration of the LOS angle. The equation of motion describing the above is:

$$T = I \cdot \ddot{\beta} \tag{3.7}$$

where:

T : Control Torque

I : Moment of Inertia

$\ddot{\beta}$: Estimated LOS angular acceleration

The input value to the filter is the LOS angle. The output value of the filter is its estimation of the angular acceleration. In between it calculates the angular velocity of the LOS. Thus solving for Equation 3.7 we get:

$$\ddot{\beta} = \frac{T}{I} = -k_1(\beta-\sigma) - k_2\dot{\beta} = -k_2\dot{\beta} - k_1\beta + k_1\sigma \qquad (3.8)$$

where k_1, k_2 are constants determined by the time constant used by the tracker. Laplace transformation transfers Equation 3.8 from the time domain to the s-domain. Thus Equation 3.8 transformed evaluates the filter transfer function:

$$\frac{\beta(s)}{\sigma(s)} = \frac{1}{(s^2+k_2 s+k_1)} \qquad (3.9)$$

Assume the relationship:

$$\frac{1}{(s^2+k_2 s+k_1)} = \frac{1}{\left(s+\frac{1}{\tau_{tr}}\right)^2} \qquad (3.10)$$

For the system a time constant of 0.1 second was selected, since it approximates current technology. So the constants can be defined:

$$k_1 = \left(\frac{1}{\tau_{tr}}\right)^2 = 100$$
$$k_2 = 2\left(\frac{1}{\tau_{tr}}\right) = 20 \qquad (3.11)$$

The block diagram and S.F.G. of the filter is shown in Figure 11.

From Figure 11 it can be seen that two integrators are incorporated in the design. We can further analyze the system

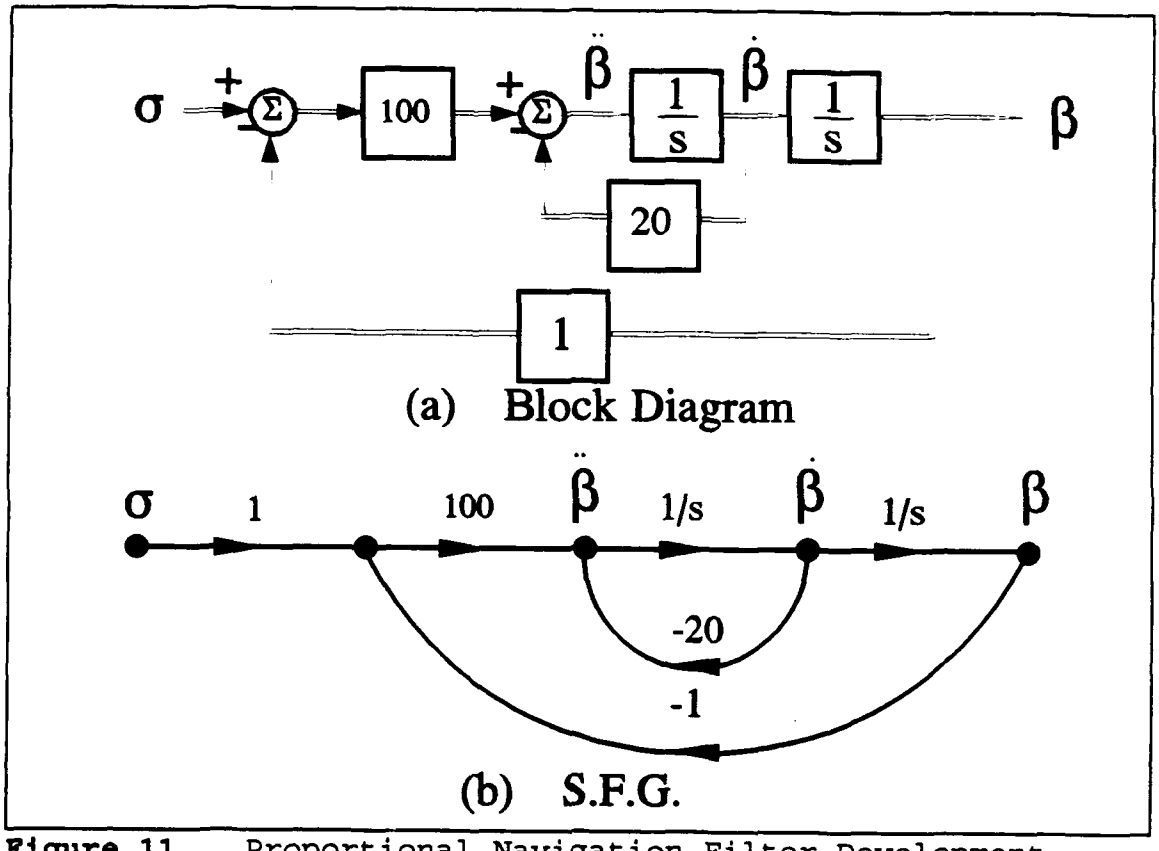


Figure 11. Proportional Navigation Filter Development

in state-variable form to be described by the following:

$$\dot{x}_{flt} = A_{flt}x_{flt} + B_{flt}u_{flt} \qquad (3.12)$$

where the control input matrix, for our scenario, is:

$$u_{flt} = \begin{bmatrix} \sigma_{pitch} \\ \sigma_{yaw} \end{bmatrix} \qquad (3.13)$$

The vector contains the pitch and yaw plane estimations of the angle and its rate. Thus the vector can be written as:

$$x_{flt} = \begin{bmatrix} \beta_{pitch} \\ \beta_{pitch} \\ \beta_{yaw} \\ \beta_{yaw} \end{bmatrix} \tag{3.14}$$

The filter A and B matrices can then be written as:

$$A_{flt} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -100 & -20 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -100 & -20 \end{bmatrix} \quad B_{flt} = \begin{bmatrix} 0 & 0 \\ 100 & 0 \\ 0 & 0 \\ 0 & 100 \end{bmatrix} \tag{3.15}$$

This representation is from the continuous time domain, as it was transferred to the s-domain. Eventually it will be needed to descritize this, in order to run the computer simulation.

2. CLOS

It was shown previously, Chapter II Section D, that the cross range error, in a two-dimensional configuration, can be evaluated as:

$$\dot{CRE} = R_m \sin(\sigma_t - \sigma_m) \tag{2.15}$$

It was also stated that this off-beam error will be used as an acceleration command.

In order for good response characteristics to be obtained, some damping is required. A dynamic equation of the form :

$$(C\ddot{R}E) \doteq a_{cmd} = K_1 \cdot (C\dot{R}E) + K_2 \cdot (CRE) \qquad (3.16)$$

needs to be satisfied. The error is thus filtered [Ref. 7: pp.12]. Laplace transformation transfers Equation 3.16 from the time domain to the s-domain. Thus Equation 3.16 transformed evaluates the filter transfer function:

$$\frac{CRE(s)}{a_{cmd}(s)} = \frac{1}{(k_1 s + k_2)} \qquad (3.17)$$

The filter gain constants (K_1, K_2) assumed here are:

$$\begin{aligned} K_1 &= -25 \\ K_2 &= -100 \end{aligned} \qquad (3.18)$$

These values are also representative of the current technology.

The block diagram and S.F.G. of the filter is shown in Figure 12. From this it can be seen that two integrators are incorporated in the design. We can further analyze the system in state-variable form to be described by the following:

$$\dot{x}_{flt} = A_{flt} x_{flt} \qquad (3.19)$$

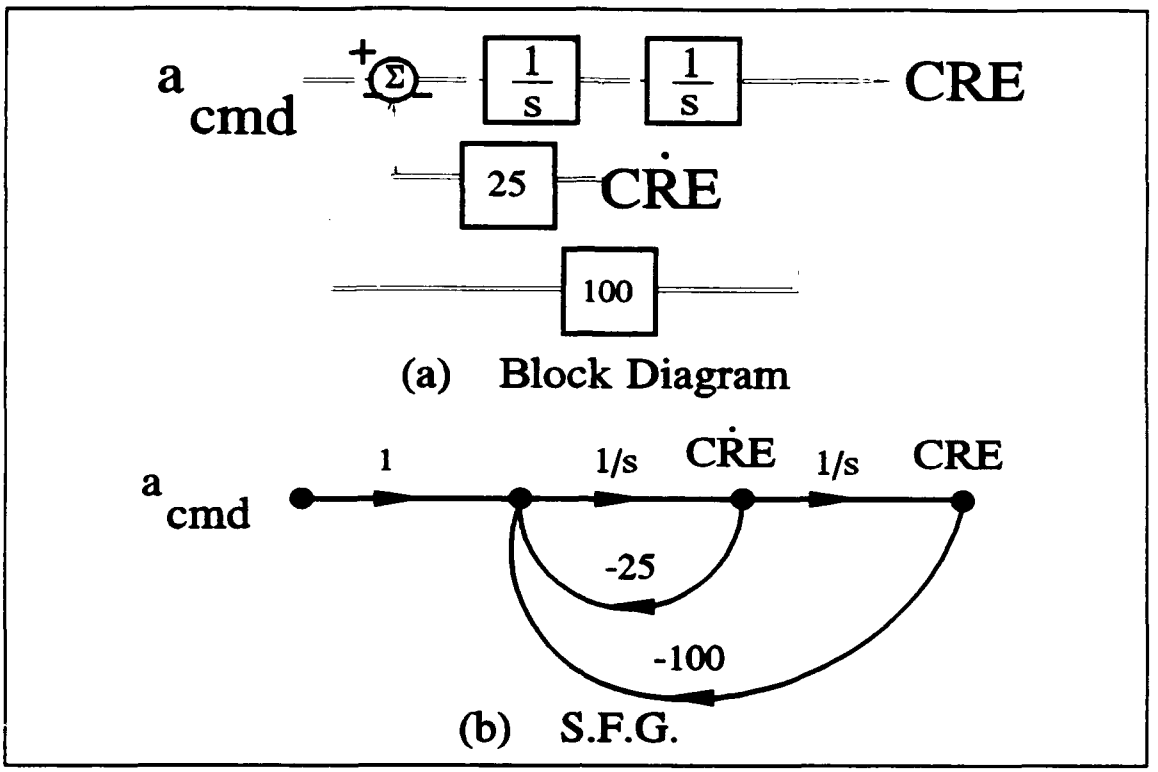


Figure 12. CLOS Filter Development

The vector contains the pitch and yaw plane estimations of the angle and its rate. Thus the vector can be written as:

$$x_{flt} = \begin{bmatrix} CRE_{pitch} \\ \dot{CRE}_{pitch} \\ CRE_{yaw} \\ \dot{CRE}_{yaw} \end{bmatrix} \tag{3.20}$$

The filter A matrix can then be written as:

$$A_{flt} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ -100 & -25 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & -100 & -25 \end{bmatrix} \tag{3.21}$$

This representation is from the continuous time domain, as it was transferred to the s-domain. Eventually it will be needed to descritize this, in order to run the computer simulation.

D. GUIDANCE DEVELOPMENT

1. Proportional Navigation

The guidance subsystem must follow the proportional navigation law, as this was stated in Equation 2.13. The only difference would be that the estimation of the angular LOS rate is used instead of the actual LOS rate, since this is the input from the filter subsystem. Thus:

$$\dot{\gamma}_m = N\hat{\beta} \tag{3.22}$$

This is also the control input transmitted to the missile's autopilot.

The state variable representation, for our case, is:

$$x_{gui} = N_{gui}u_{gui} \rightarrow$$

$$\begin{bmatrix} \dot{\gamma}_{m_{pitch}} \\ \dot{\gamma}_{m_{yaw}} \end{bmatrix} = \begin{bmatrix} N & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} \hat{\beta}_{pitch} \\ \hat{\beta}_{yaw} \end{bmatrix} \tag{3.23}$$

2. CLOS

A generic CLOS guidance would generate a missile acceleration equal to the estimated acceleration of the cross range error;i.e.:

$$a_m = a_{cmd} \tag{3.24}$$

This will also be the control input transmitted to the missile's autopilot.

The state variable representation, for this case, is:

$$\begin{bmatrix} a_{cmd_{pitch}} \\ a_{cmd_{yaw}} \end{bmatrix} = \begin{bmatrix} -25 & -100 \\ 0 & 0 \end{bmatrix} \begin{bmatrix} CRE_{pitch} \\ C\dot{R}E_{pitch} \end{bmatrix} + \begin{bmatrix} 0 & 0 \\ -25 & -100 \end{bmatrix} \begin{bmatrix} CRE_{yaw} \\ C\dot{R}E_{yaw} \end{bmatrix} \tag{3.25}$$

E. AUTOPILOT DEVELOPMENT

1. Proportional Navigation

The autopilot determines the control (actuator position and thrust) necessary to perform the required command. In proportional navigation guidance, the missile commands are generated in order to change the missile flight path rate in proportion to the LOS rate.

A simplified autopilot, would respond to the following argument [Ref. 5: pp.17-19]. The applied torque about the missile center of gravity is proportional to the angular acceleration of the missile flight path. This, equation of motion , stated mathematically yields:

$$T_{app} = I_{cg} \ddot{\gamma}_m \tag{3.26}$$

where:

T_{app} : The applied torque

I_{cg} : The moment of inertia around the missile's center of gravity

γ_m : The angular acceleration of the missile's flight path

The control torque, discussed in Chapter III Section C, may be different from the applied torque. Thus solving Equation 3.26 for the flight path acceleration:

$$\ddot{\gamma}_m = \frac{T_{app}}{I_{cg}} = -k\dot{\gamma}_m + kN\beta \qquad (3.27)$$

where k is determined by the slowest time constant of the missile/autopilot. Laplace transformation transfers Equation 3.27 from the time domain to the s-domain. Thus Equation 3.27 transformed evaluates the autopilot transfer function:

$$\frac{\dot{\gamma}_m(s)}{\beta(s)} = \frac{kN}{s+k} \qquad (3.28)$$

For the autopilot a time constant of 1.0 second was selected. So the constant can be defined:

$$k = \frac{1}{\tau_{ap}} = 1 \qquad (3.29)$$

The autopilot block diagram and S.F.G. are shown in Figure 13. From this it can be seen that although two integrators are incorporated in the design, only one output is fedback. We can further analyze the system in state-variable

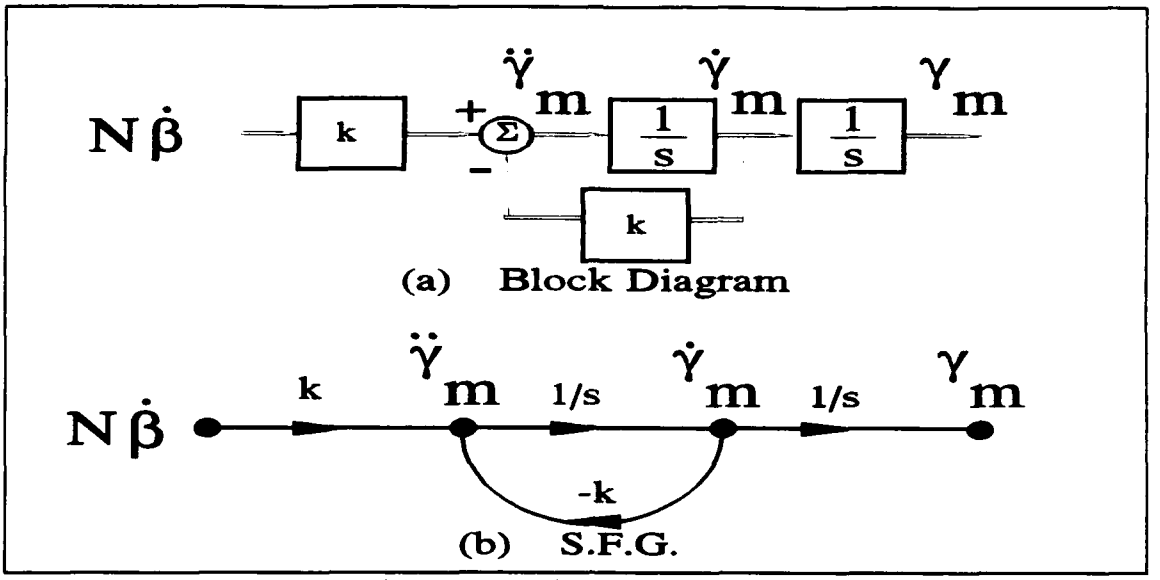


Figure 13. Proportional Navigation Autopilot Development

form to be described by the following:

$$\dot{X}_{ap} = A_{ap}X_{ap} + B_{ap}U_{ap} \qquad (3.30)$$

The vector contains the pitch and yaw plane flight path angular velocity estimations. Thus the vector can be written as:

$$X_{ap} = \begin{bmatrix} \dot{\gamma}_{m_{pitch}} \\ \dot{\gamma}_{m_{yaw}} \end{bmatrix} \qquad (3.31)$$

The autopilot A and B matrices can then be written as:

$$A_{ap} = \begin{bmatrix} -1 & 0 \\ 0 & -1 \end{bmatrix} \qquad B_{ap} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad (3.32)$$

The control input matrix is:

$$u_{ap} = B_{gui}\, u_{gui} = \begin{bmatrix} N & 0 \\ 0 & N \end{bmatrix} \begin{bmatrix} \beta_{pitch} \\ \beta_{yaw} \end{bmatrix} \tag{3.33}$$

This representation is from the continuous time domain, as it was transferred to the s-domain. Eventually it will be needed to descritize this, in order to run the computer simulation.

This direction change rate is converted to an acceleration command to the actuators. First, the missile velocity must be analyzed in the pitch and yaw plane. Referring to Figure 14, the missile velocity vector components can be stated as follows:

$$\begin{aligned} V_{m_{pitch}} &= \left| V_m \cos\left(\gamma_{m_{yaw}} - \sigma_{m_{yaw}}\right) \right| \\ V_{m_{yaw}} &= \left| V_m \cos\gamma_{m_{pitch}} \right| \end{aligned} \tag{3.34}$$

Then the acceleration components can be derived:

$$\begin{aligned} a_{m_{pitch}} &= V_{m_{pitch}}\, \dot{\gamma}_{m_{pitch}} \\ a_{m_{yaw}} &= V_{m_{yaw}}\, \dot{\gamma}_{m_{yaw}} \end{aligned} \tag{3.35}$$

The spatial accelerations are the same for the CLOS autopilot and are derived in the following Subsection 2.

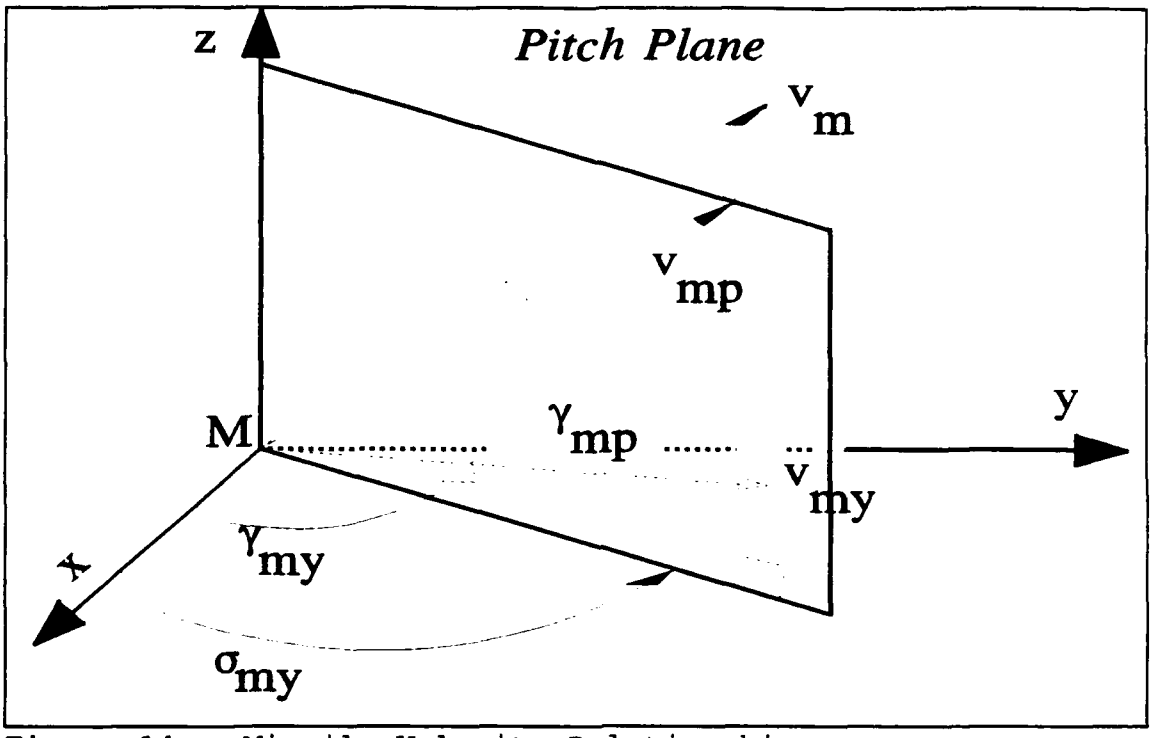


Figure 14. Missile Velocity Relationships

2. CLOS

The acceleration commanded is distributed by the autopilot into the missile's spatial accelerations. Referring to Figure 15:

$$\begin{aligned} \ddot{x}_{m_{pitch}} &= -a_{m_{pitch}} \sin\sigma_{pitch} \cos\sigma_{yaw} \\ \ddot{y}_{m_{pitch}} &= -a_{m_{pitch}} \sin\sigma_{pitch} \sin\sigma_{yaw} \\ \ddot{z}_{m_{pitch}} &= -a_{m_{pitch}} \cos\sigma_{pitch} \end{aligned} \tag{3.36}$$

and, similarly, for the yaw plane:

$$\begin{aligned} \ddot{x}_{m_{yaw}} &= -a_{m_{yaw}} \sin\sigma_{yaw} \\ \ddot{y}_{m_{yaw}} &= -a_{m_{yaw}} \cos\sigma_{yaw} \end{aligned} \tag{3.37}$$

From the above relationships, the autopilot states can be defined. These states, as well as those for proportional

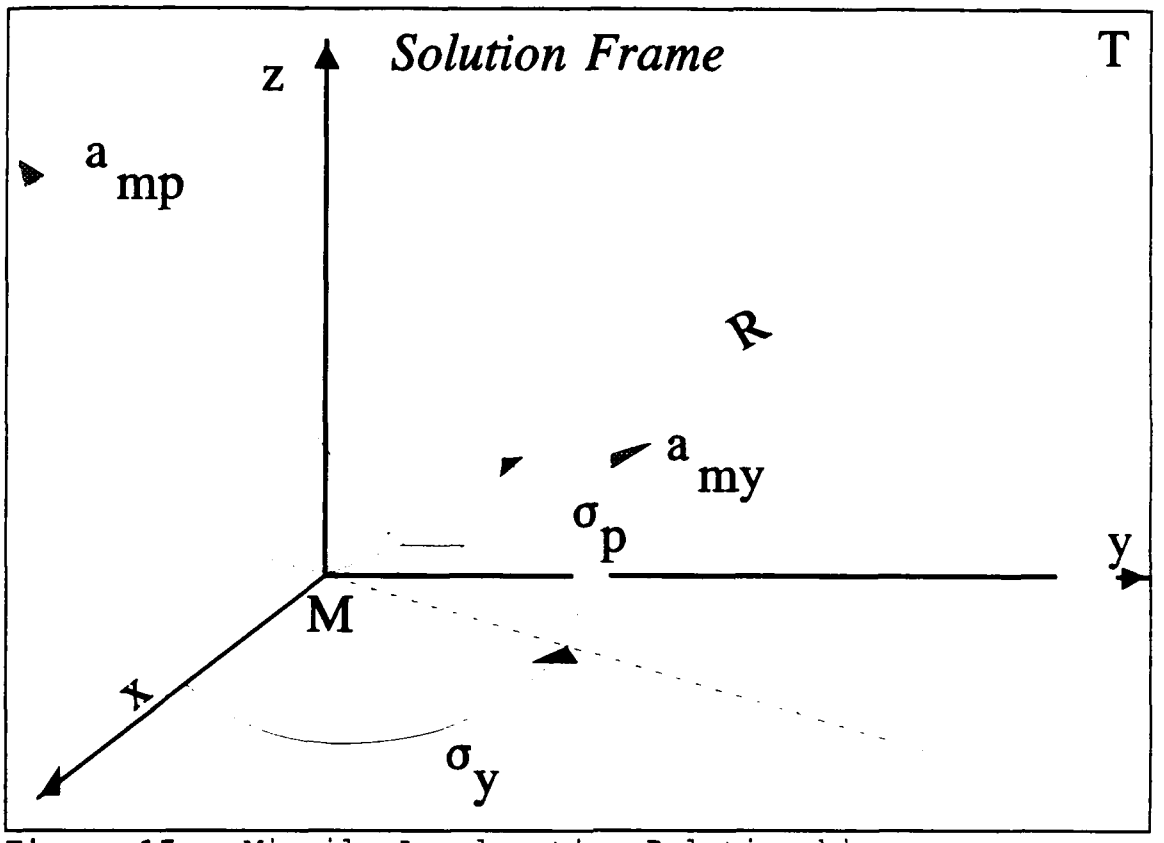


Figure 15. Missile Acceleration Relationships

navigation guidance, are the control inputs for the missile airframe. So, the following is concluded:

$$\begin{aligned} \ddot{x}_m &= \ddot{x}_{m_{pitch}} + \ddot{x}_{m_{yaw}} \\ \ddot{y}_m &= \ddot{y}_{m_{pitch}} + \ddot{y}_{m_{yaw}} \\ \ddot{z}_m &= \ddot{z}_{m_{pitch}} \end{aligned} \tag{3.38}$$

F. ACTUATOR DEVELOPMENT

In the present model, and for both guidance cases, the assumption is made that the actuators, also known as control servos, directly convert the commanded rate to fin

deflections. This being the case, the transfer function is unity.

G. AIRFRAME DEVELOPMENT

The missile vector is defined as the missile's spatial and velocity components, namely:

$$x_m = \begin{bmatrix} x_m \\ \dot{x}_m \\ y_m \\ \dot{y}_m \\ z_m \\ \dot{z}_m \end{bmatrix} \tag{3.39}$$

Thus the missile can be represented as:

$$\dot{x}_m = A_m x_m + B_m u_m \tag{3.40}$$

The control input matrix is:

$$u_m = \begin{bmatrix} \ddot{x}_m \\ \ddot{y}_m \\ \ddot{z}_m \end{bmatrix} \tag{3.41}$$

Thus the A and B matrices can be written as follows:

$$A_m = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad B_m = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.42}$$

A S.F.G. representing the missile dynamics for the y-axis and for both guidances is shown in Figure 16.

H. KINEMATICS DEVELOPMENT

The airframe development also defines the missile kinematics. The target kinematics are derived in a similar fashion. The target can be described by a state equation, namely:

$$\dot{x}_t = A_t x_t + B_t u_t \tag{3.43}$$

where:

$$x_t = \begin{bmatrix} x_t \\ \dot{x}_t \\ y_t \\ \dot{y}_t \\ z_t \\ \dot{z}_t \end{bmatrix} \tag{3.44}$$

$$A_t = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix} \qquad B_t = \begin{bmatrix} 0 & 0 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix} \tag{3.45}$$

$$u_t = \begin{bmatrix} \ddot{x}_t \\ \ddot{y}_t \\ \ddot{z}_t \end{bmatrix} \tag{3.46}$$

The target dynamic equations are defined with respect to the trajectory it follows; for example if the target is at level flight the accelerations are zero and so is the control matrix.

The S.F.G. for the target kinematics, for the y-axis, is shown in Figure 17.

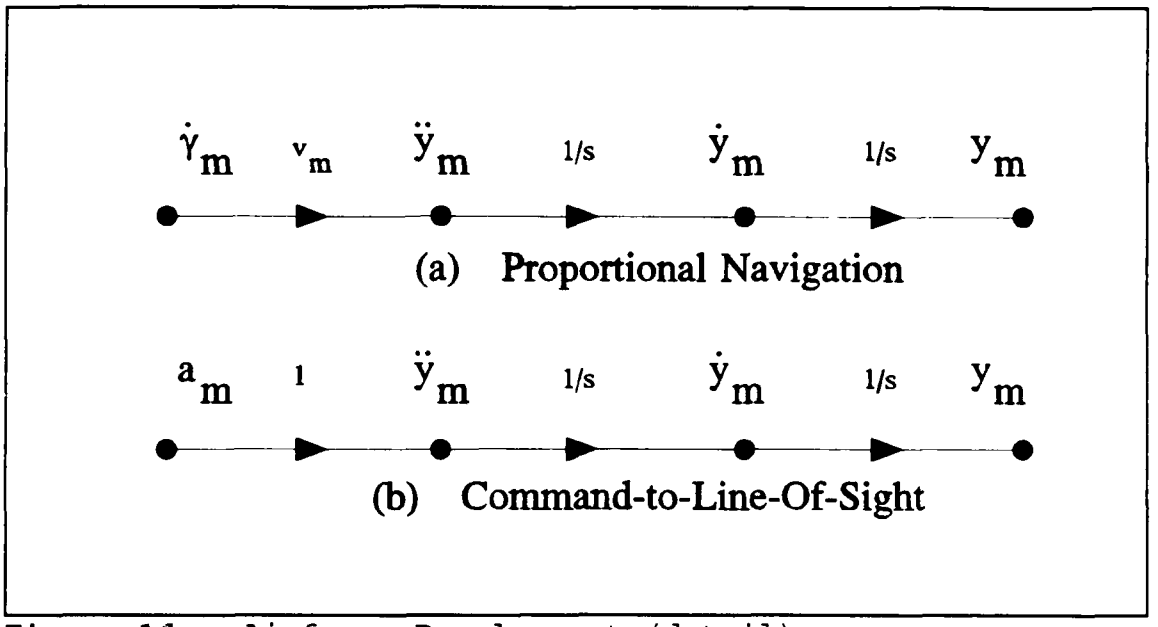


Figure 16. Airframe Development (detail)

I. SIMULATION DEVELOPMENT

In order to simulate the system some variables are required for checking the results and the validity of the assumptions. Also the discretization of the system needs to be addressed.

1. Additional Calculations

The missile-to-target range (R) can also be defined as:

$$R = v_c \, t_{go} \tag{3.47}$$

where:

v_c : the closing velocity

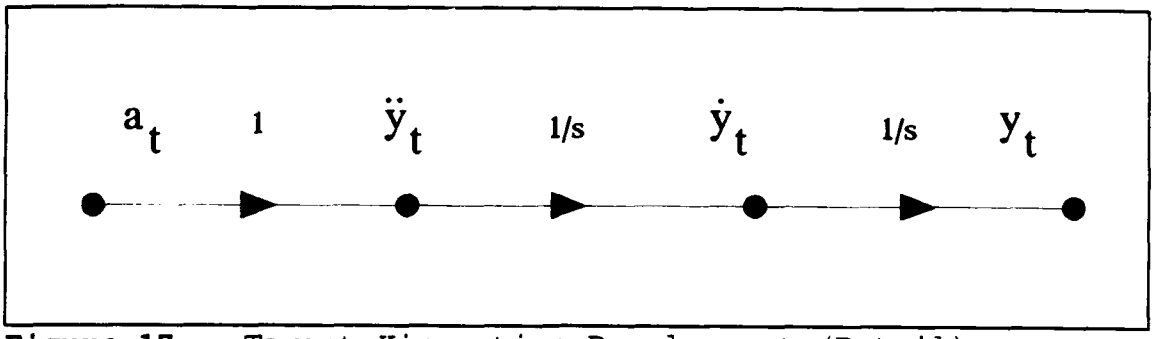


Figure 17. Target Kinematics Development (Detail)

t_{go} : the time-to-go until impact;i.e. the initial value must equal the total flight time (t_{CPA})

The closing velocity was defined in Equation 2.3. Referring to Figure 18:

$$V_c = -\dot{R} = V_{t_{yaw}}\cos(\gamma_{t_{yaw}} - \sigma_{t_{yaw}}) - V_{m_{yaw}}\cos(\gamma_{m_{yaw}} - \sigma_{yaw}) \quad \textbf{(3.48)}$$

A similar analysis can be done in the pitch plane, but the idea is to subtract the missile and target velocity components parallel to the projection of the miss distance in either the yaw or the pitch plane.

Thus the time-to-go can be calculated as:

$$t_{go} = \frac{R}{-\dot{R}} = \frac{R}{V_c} \quad \textbf{(3.49)}$$

2. System Discretization

For a continuous time state-space system described by the following set of equations [Ref. 8]:

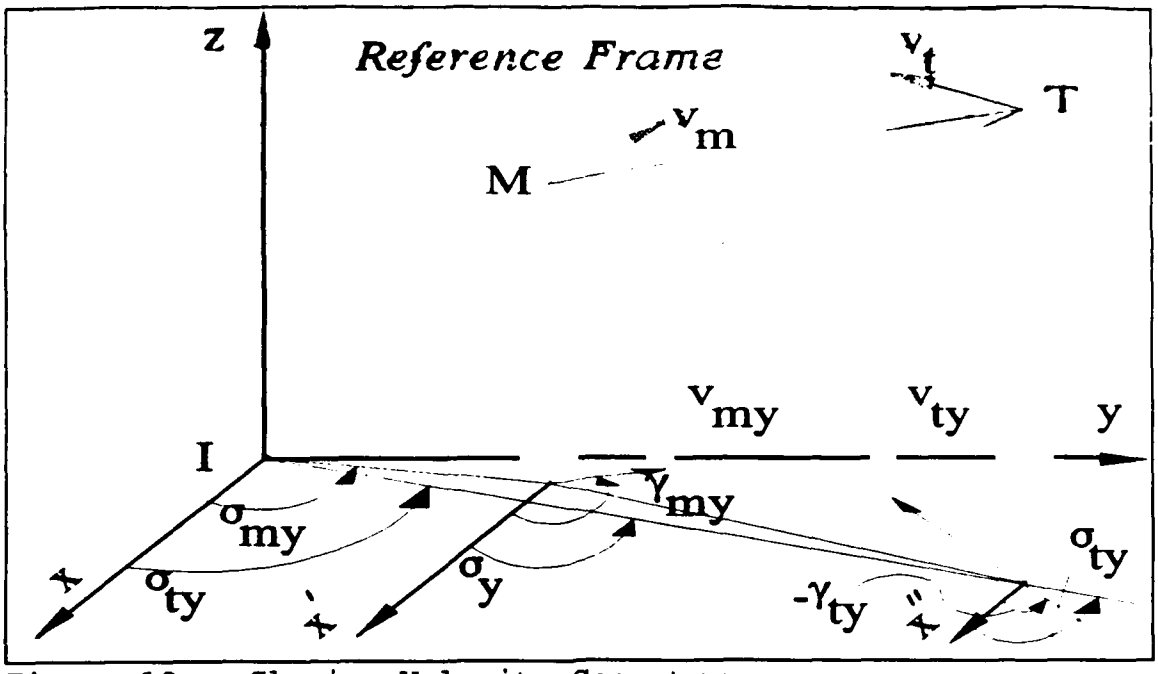


Figure 18. Closing Velocity Geometry

$$\begin{aligned} \dot{x}(t) &= Ax(t) + Bu(t) \\ y(t) &= Cx(t) + Du(t) \end{aligned} \tag{3.50}$$

a transformation into a discrete time system, with a T time interval, can be achieved:

$$\begin{aligned} x(k+1) &= \Phi x(k) + \Gamma u(k) \\ y(k) &= Cx(k) + Du(k) \end{aligned} \tag{3.51}$$

according to the transformation equations:

$$\begin{aligned} \Phi &= e^{AT} \\ \Gamma &= \left[\int_0^T e^{At}dt\right] B \end{aligned} \tag{3.52}$$

One way of calculating the matrix exponent is:

$$e^{AT} = \mathcal{L}^{-1}[(sI-A)^{-1}]\Big|_{t=T}$$
$$= I+AT+\frac{A^2T^2}{2!}+\frac{A^3T^3}{3!}+\ldots \qquad (3.53)$$

For the simulation the MATLAB function C2D.M is utilized. This function produces the Φ and Γ when the A and B matrices as well as the time interval (T) are specified.

IV. SIMULATION RESULTS

A. OVERVIEW

The computer code for the proportional navigation command guidance is presented in Appendix 1, and that for the command-to-line-of-sight in Appendix 2. The assumptions that have been made throughout will be discussed. The two scenarios run and their results will also be presented.

B. ASSUMPTIONS

The following assumptions have been made throughout the analysis and simulation:

- The missile is not limited in accelerations.
- The acceleration due to gravity is ignored.
- The target is capable of instantaneous acceleration.
- The target has no upper acceleration limit.
- The proportional navigation constant is 4.
- The missile is pointed to the target at launch.

C. ENGAGEMENT SCENARIOS

1. Target at Steady, Level Flight

For this scenario the target is flying at a constant altitude with no acceleration. The missile is fired from the

origin of the reference frame. The initial conditions of the missile states are:

$$\begin{aligned} x_m &= 0 && [ft] \\ \dot{x}_m &= 3000 && [ft/\text{sec}] \\ y_m &= 0 && [ft] \\ \dot{y}_m &= 0 && [ft/\text{sec}] \\ z_m &= 0 && [ft] \\ \dot{z}_m &= 0 && [ft/\text{sec}] \end{aligned} \tag{4.1}$$

The initial target states are such that the target would pass above the launching platform:

$$\begin{aligned} x_t &= 30000 && [ft] \\ \dot{x}_t &= -999.445 && [ft/\text{sec}] \\ y_t &= 1000 && [ft] \\ \dot{y}_t &= -33.315 && [ft/\text{sec}] \\ z_t &= 500 && [ft] \\ \dot{z}_t &= 0 && [ft/\text{sec}] \end{aligned} \tag{4.2}$$

The target control matrix inputs are:

$$\begin{aligned} \ddot{x}_t &= 0 && [ft/\text{sec}^2] \\ \ddot{y}_t &= 0 && [ft/\text{sec}^2] \\ \ddot{z}_t &= 0 && [ft/\text{sec}^2] \end{aligned} \tag{4.3}$$

2. Maneuvering Target

For this scenario the target is accelerating in all three directions. The missile is fired from the origin of the

reference frame. The initial conditions of the missile states are:

$$\begin{aligned} x_m &= 0 && [ft] \\ \dot{x}_m &= 3000 && [ft/\text{sec}] \\ y_m &= 0 && [ft] \\ \dot{y}_m &= 0 && [ft/\text{sec}] \\ z_m &= 0 && [ft] \\ \dot{z}_m &= 0 && [ft/\text{sec}] \end{aligned} \tag{4.4}$$

The initial target states are such that the target would pass above the launching platform, if it was at level fight:

$$\begin{aligned} x_t &= 30000 && [ft] \\ \dot{x}_t &= -999.445 && [ft/\text{sec}] \\ y_t &= 1000 && [ft] \\ \dot{y}_t &= -33.315 && [ft/\text{sec}] \\ z_t &= 900 && [ft] \\ \dot{z}_t &= 0 && [ft/\text{sec}] \end{aligned} \tag{4.5}$$

The target control matrix inputs are:

$$\begin{aligned} \ddot{x}_t &= -6.5 \cdot 32.2 \sin\gamma_{t_{yaw}} && [ft/\text{sec}^2] \\ \ddot{y}_t &= -6.5 \cdot 32.2 \cos\gamma_{t_{yaw}} && [ft/\text{sec}^2] \\ \ddot{z}_t &= -0.1 \cdot 32.2 \cos\gamma_{t_{pitch}} && [ft/\text{sec}^2] \end{aligned} \tag{4.6}$$

D. RESULTS

Figures 19 through 38 display the output of the proportional navigation code for the first scenario.

Figures 39 through 58 display the output of the proportional navigation code for the second scenario.

Figures 59 through 80 display the output of the CLOS code for the first scenario.

Figures 81 through 102 display the output of the CLOS code for the second scenario.

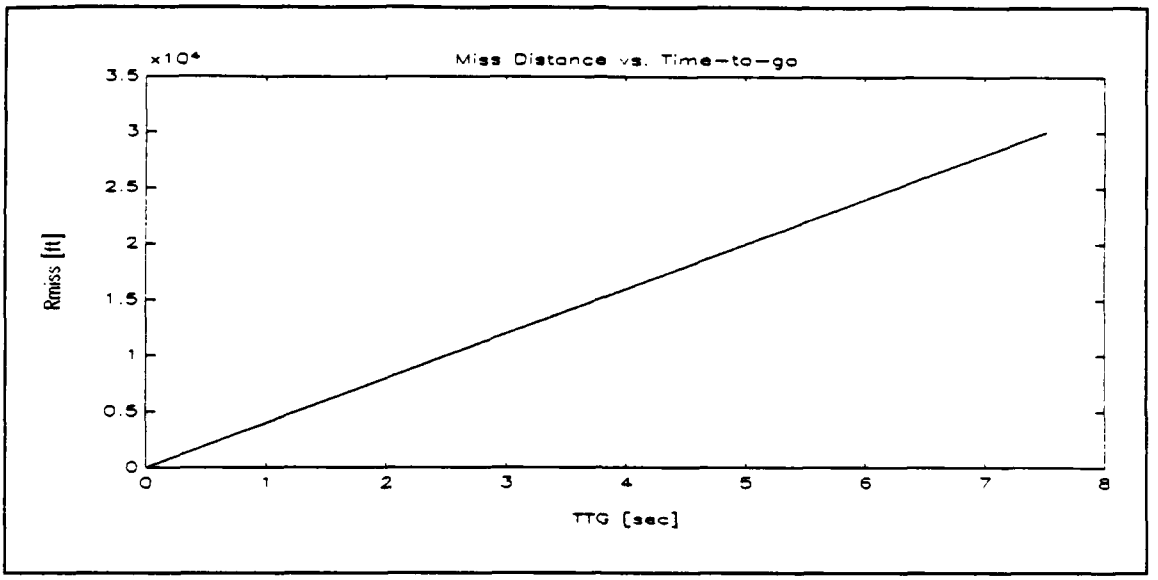


Figure 19. Proportional Navigation Scenario 1: Time-To-Go vs. Missile-to-Target Range

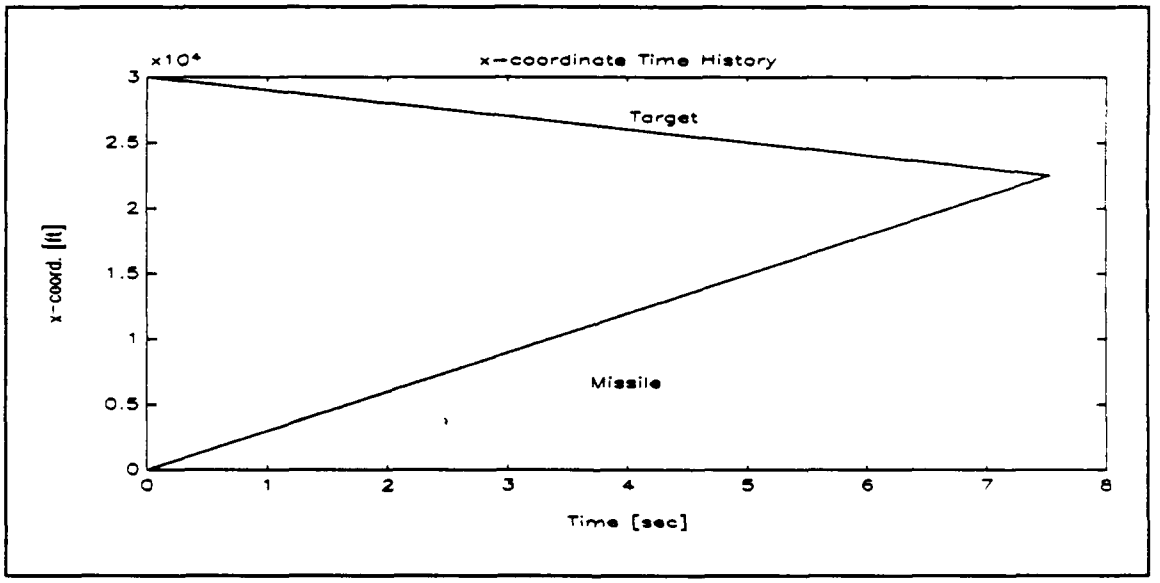


Figure 20. Proportional Navigation Scenario 1: Missile and Target x-coordinate Time History

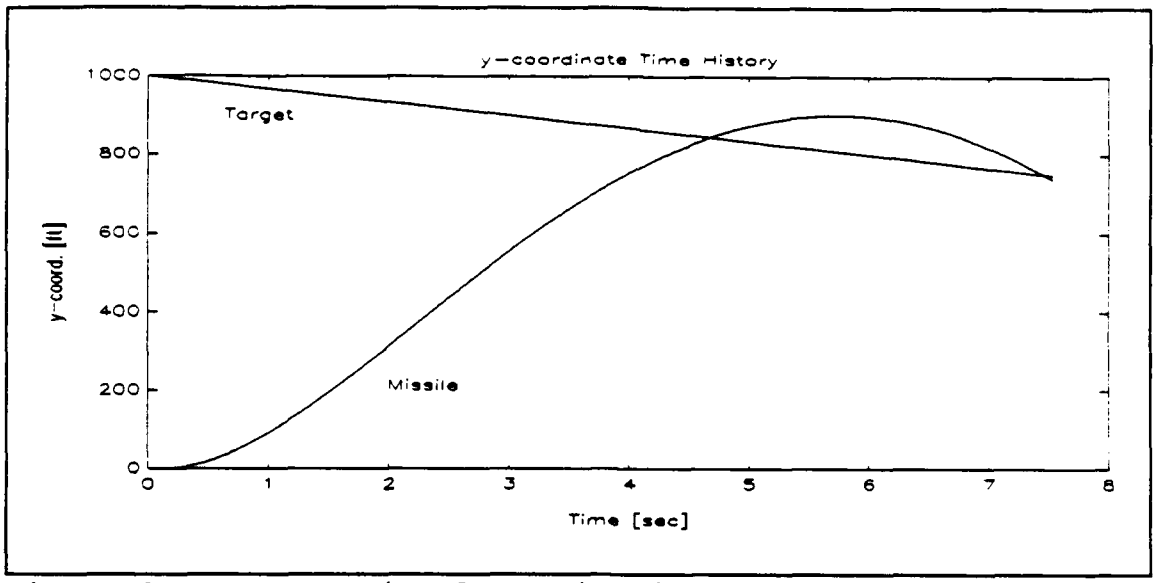


Figure 21. Proportional Navigation Scenario 1: Missile and Target y-coordinate Time History

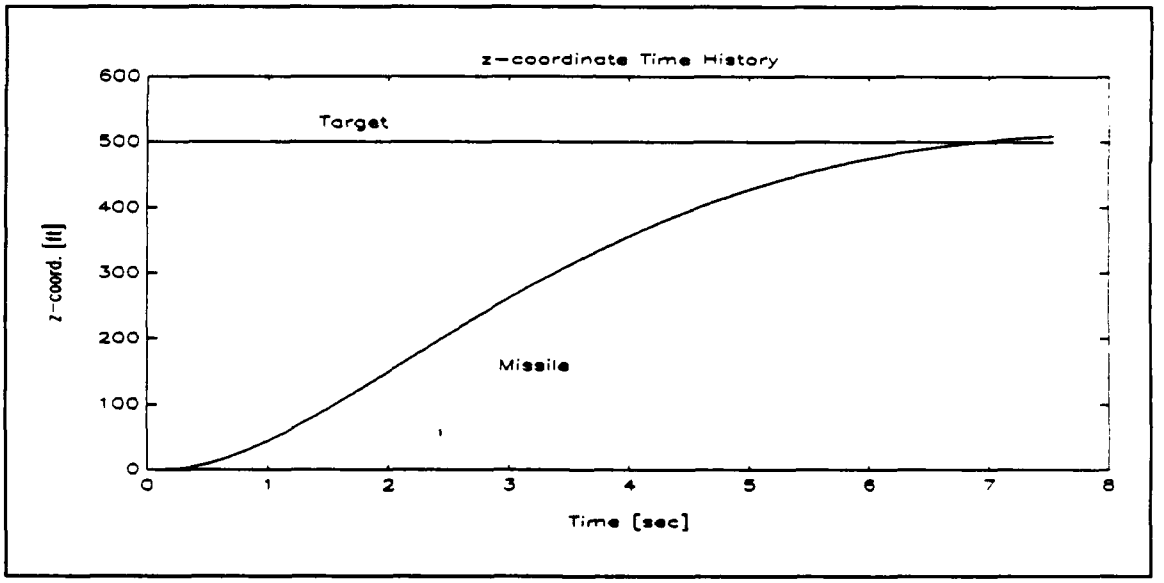


Figure 22. Proportional Navigation Scenario 1: Missile and Target z-coordinate Time History

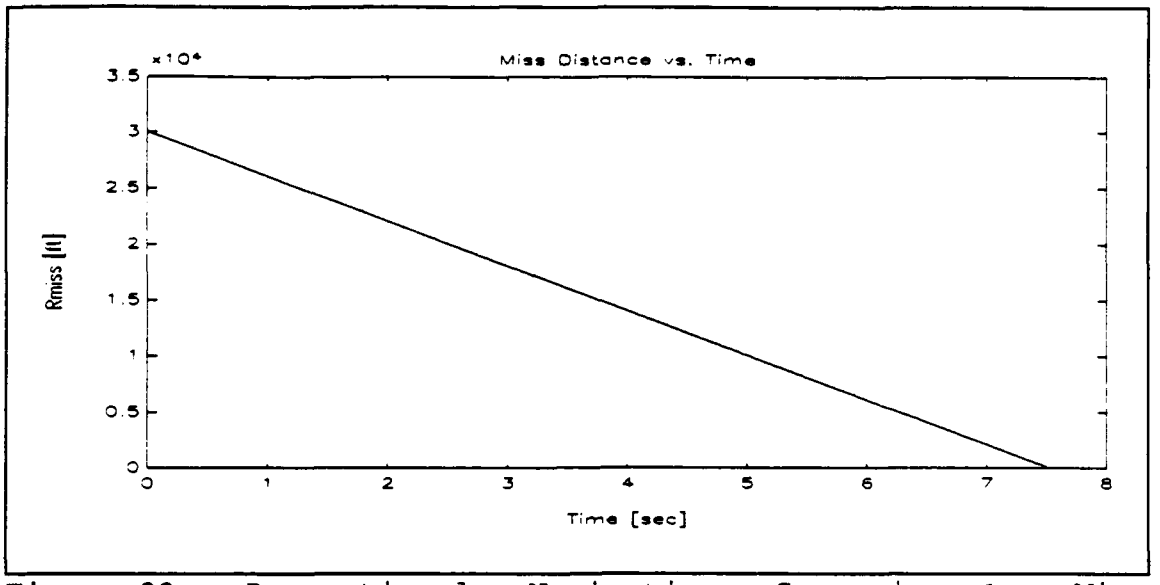


Figure 23. Proportional Navigation Scenario 1: Miss Distance Time History

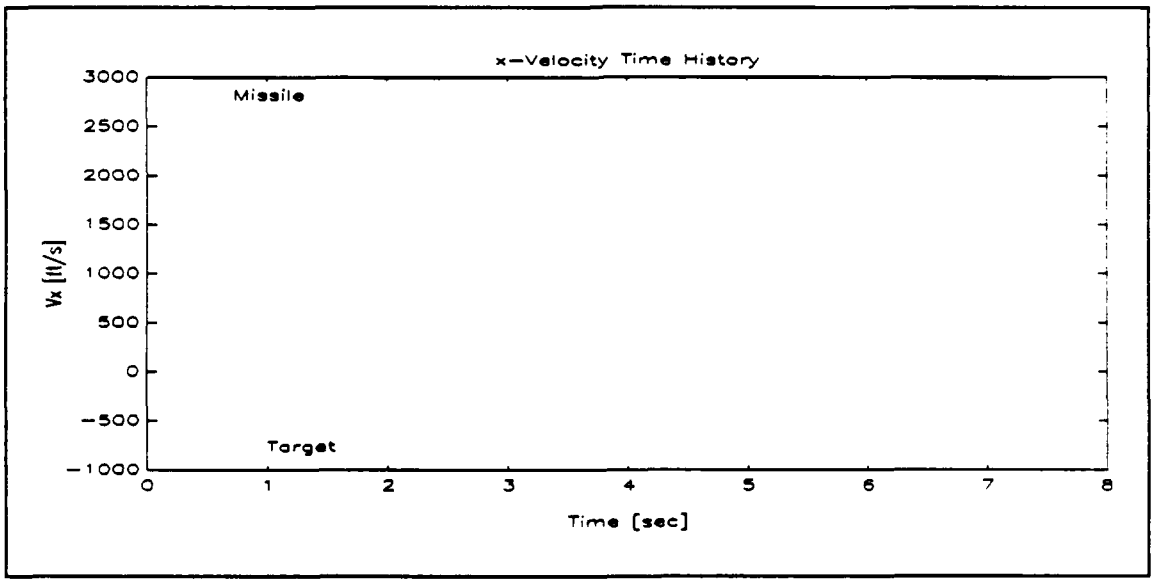


Figure 24. Proportional Navigation Scenario 1: Missile and Target Velocity (x-component) Time History

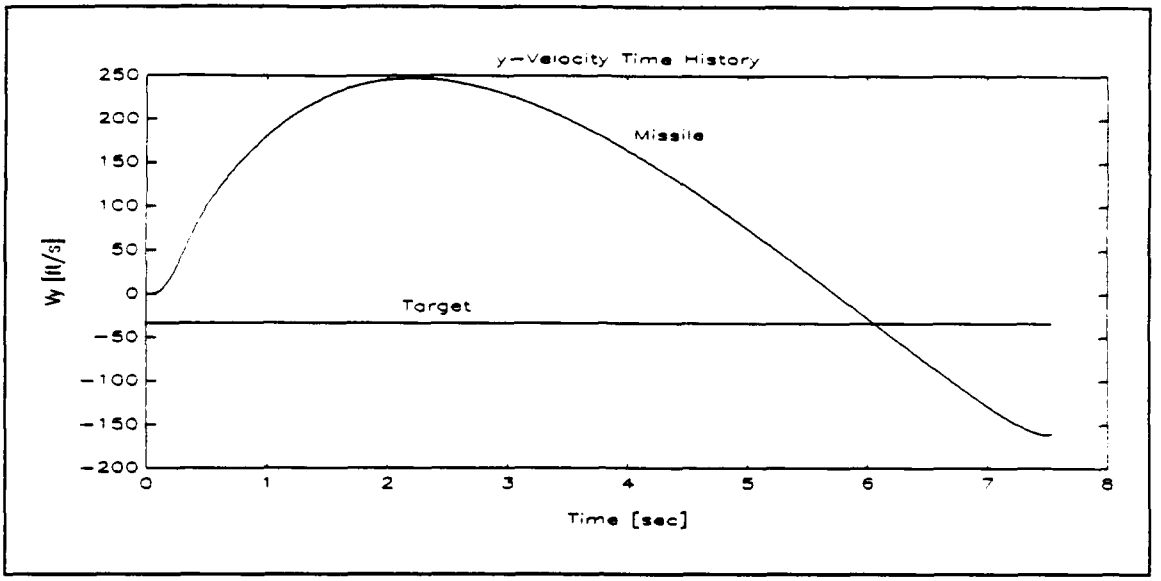


Figure 25. Proportional Navigation Scenario 1: Missile and Target Velocity (y-component) Time History

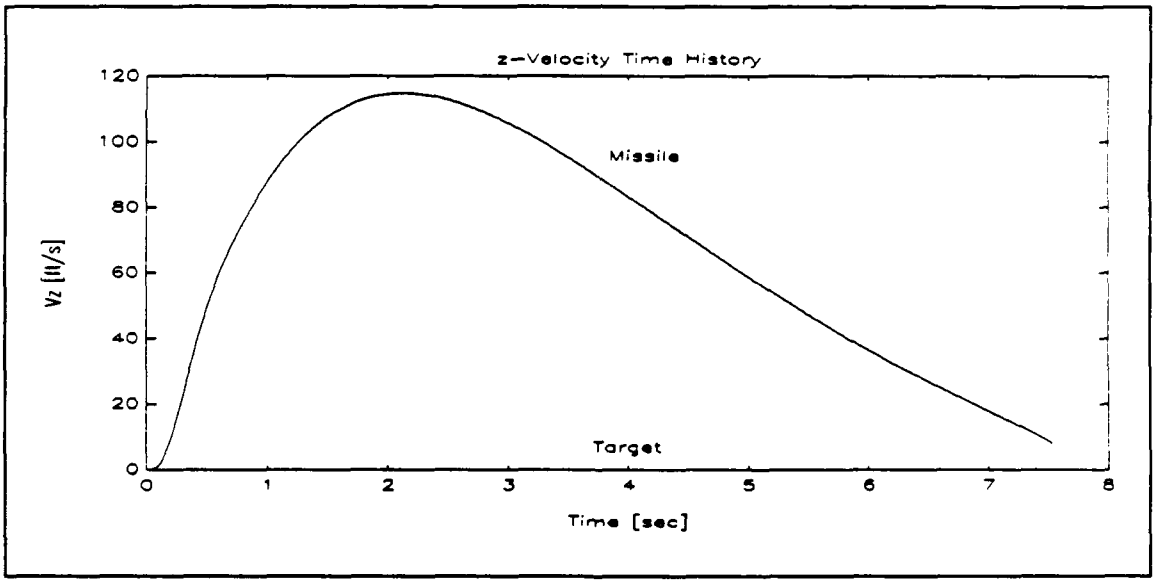


Figure 26. Proportional Navigation Scenario 1: Missile and Target Velocity (z-component) Time History

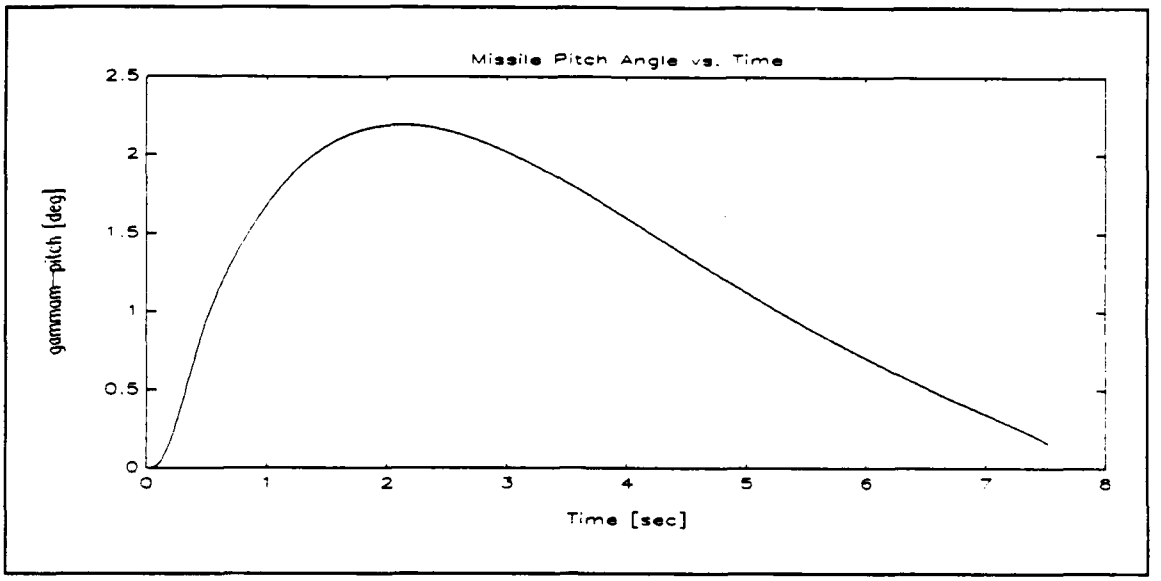


Figure 27. Proportional Navigation Scenario 1: Missile Pitch Angle Time History

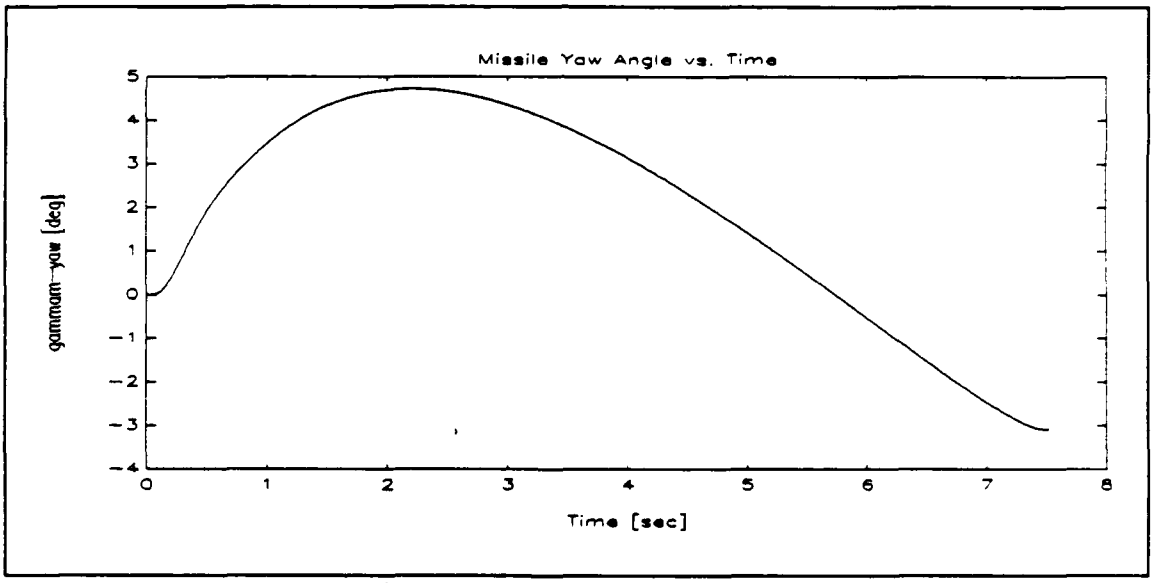


Figure 28. Proportional Navigation Scenario 1: Missile Yaw Angle Time History

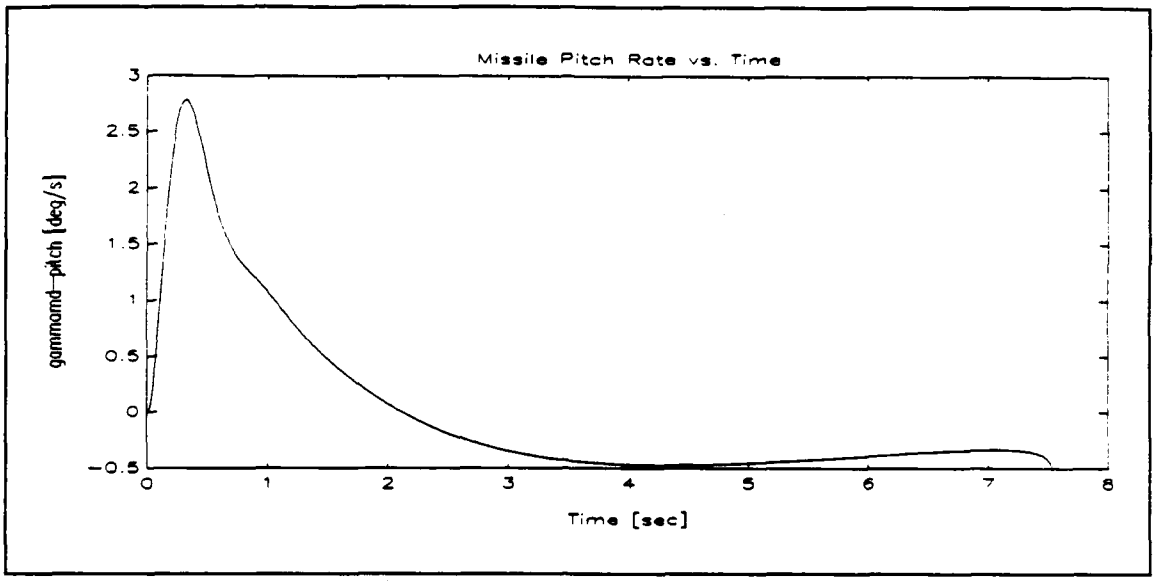


Figure 29. Proportional Navigation Scenario 1: Missile Pitch Rate Time History

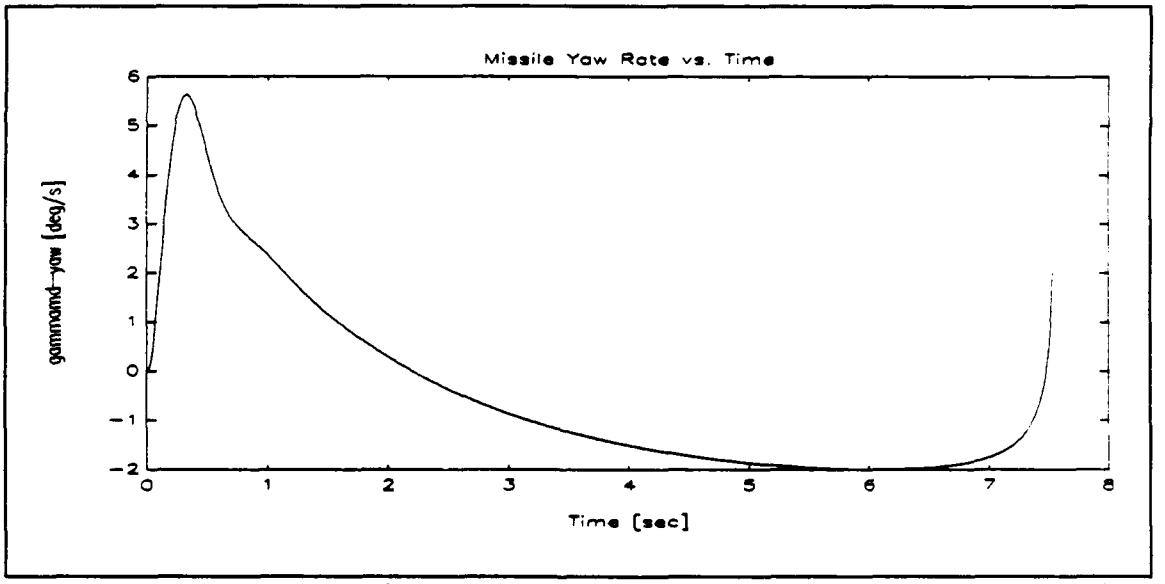


Figure 30. Proportional Navigation Scenario 1: Missile Yaw Rate Time History

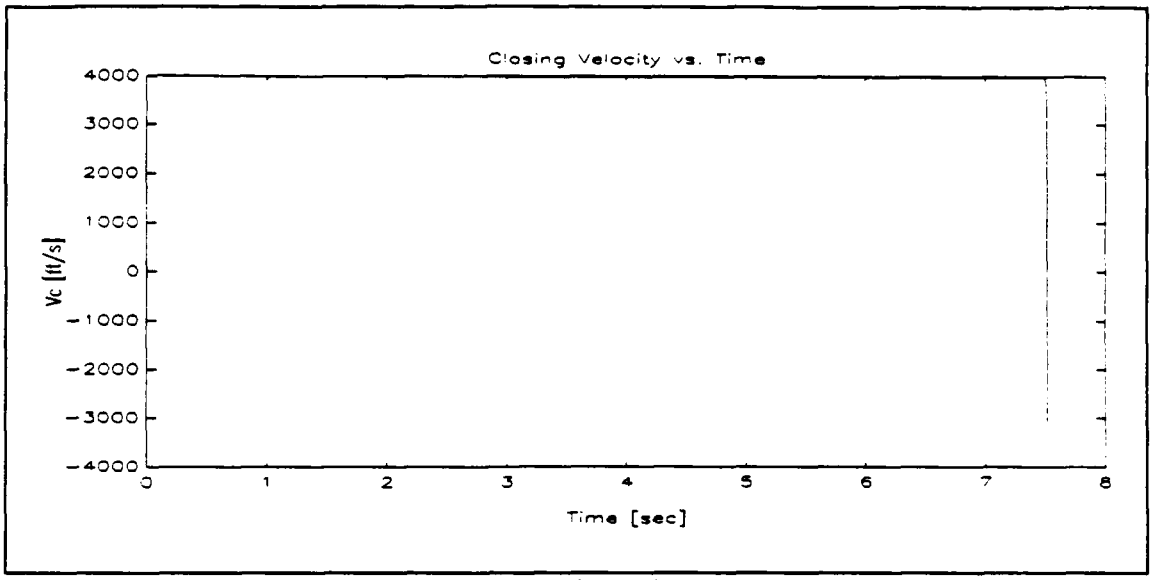


Figure 31. Proportional Navigation Scenario 1: Closing Velocity Time History

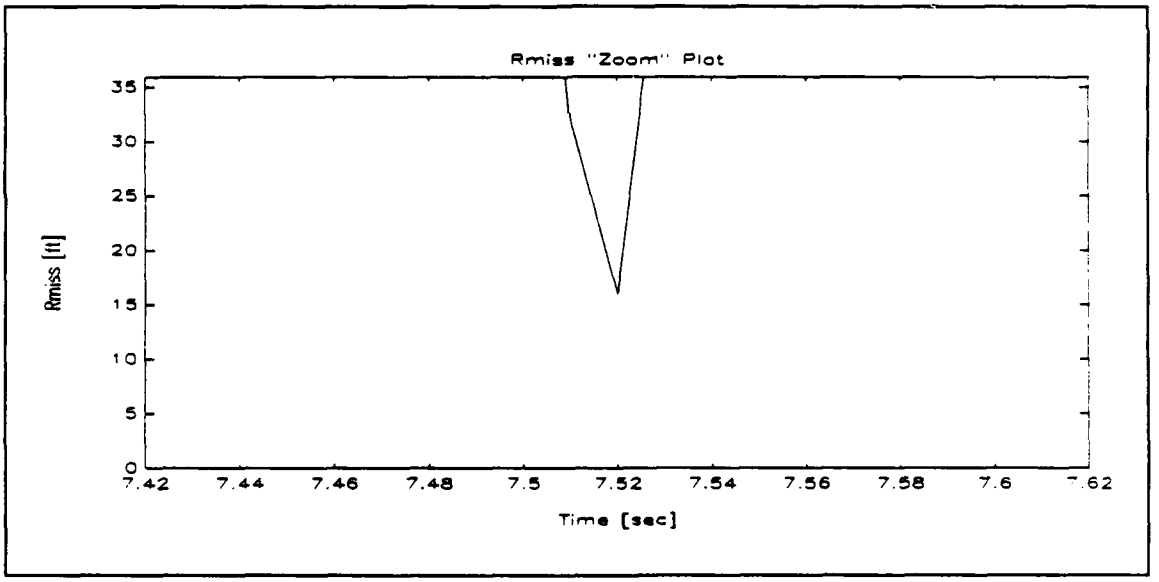


Figure 32. Proportional Navigation Scenario 1: Miss Distance and Time-of-Closest-Point-of Approach Enhancement Plot

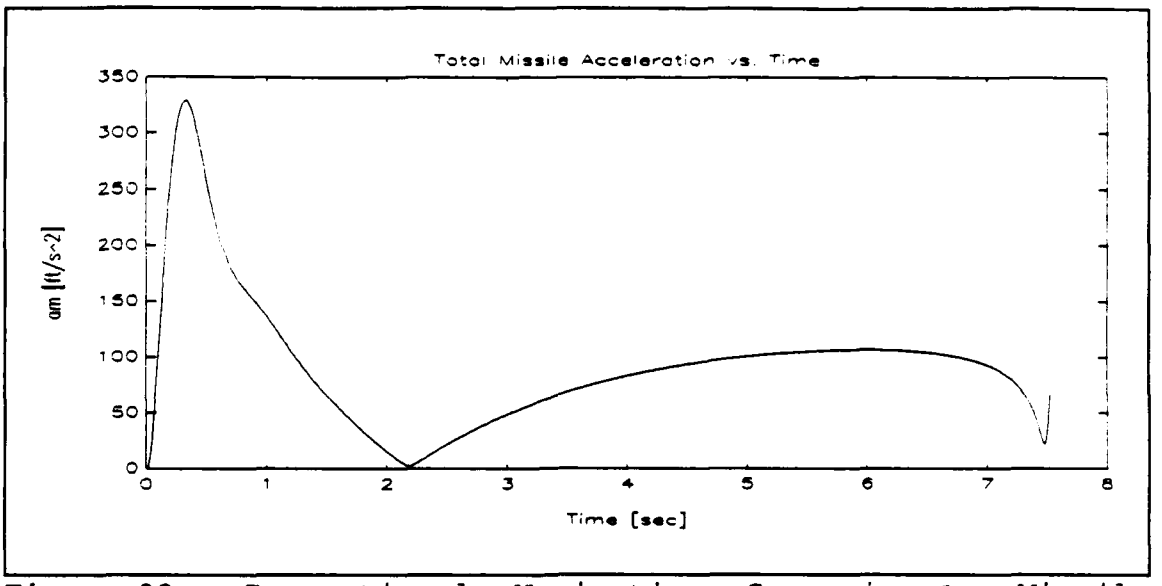


Figure 33. Proportional Navigation Scenario 1: Missile Total Acceleration Time History

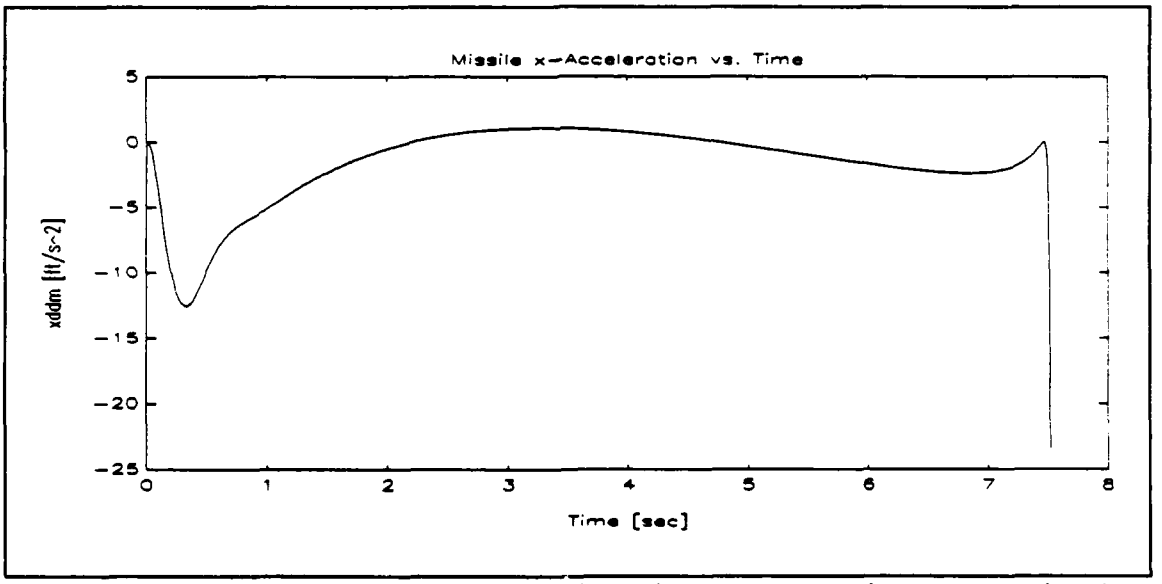


Figure 34. Proportional Navigation Scenario 1: Missile Acceleration (x-component) Time History

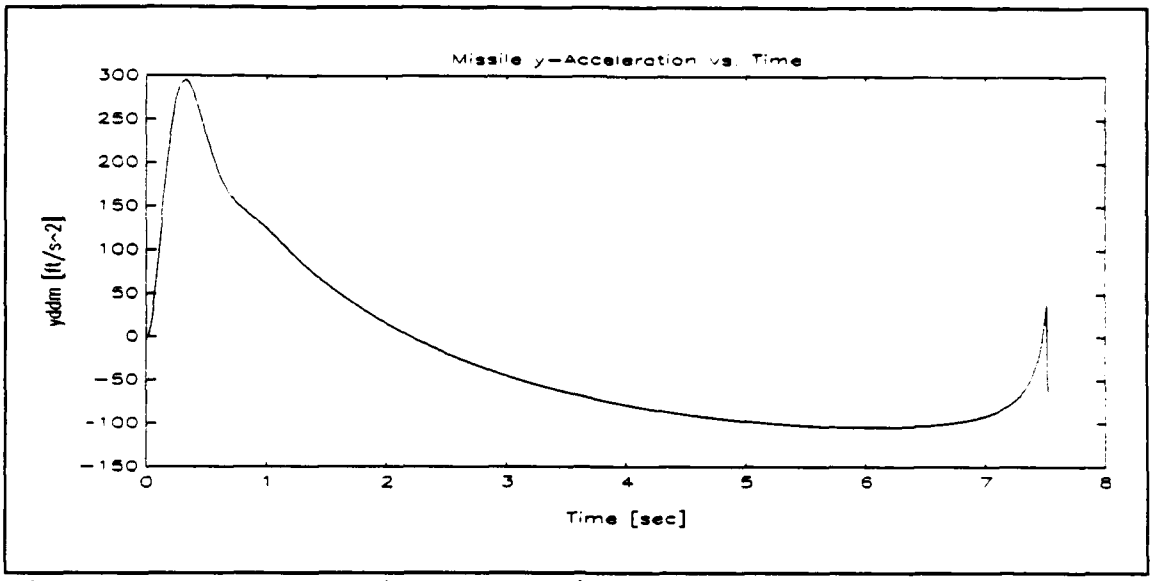


Figure 35. Proportional Navigation Scenario 1: Missile Acceleration (y-component) Time History

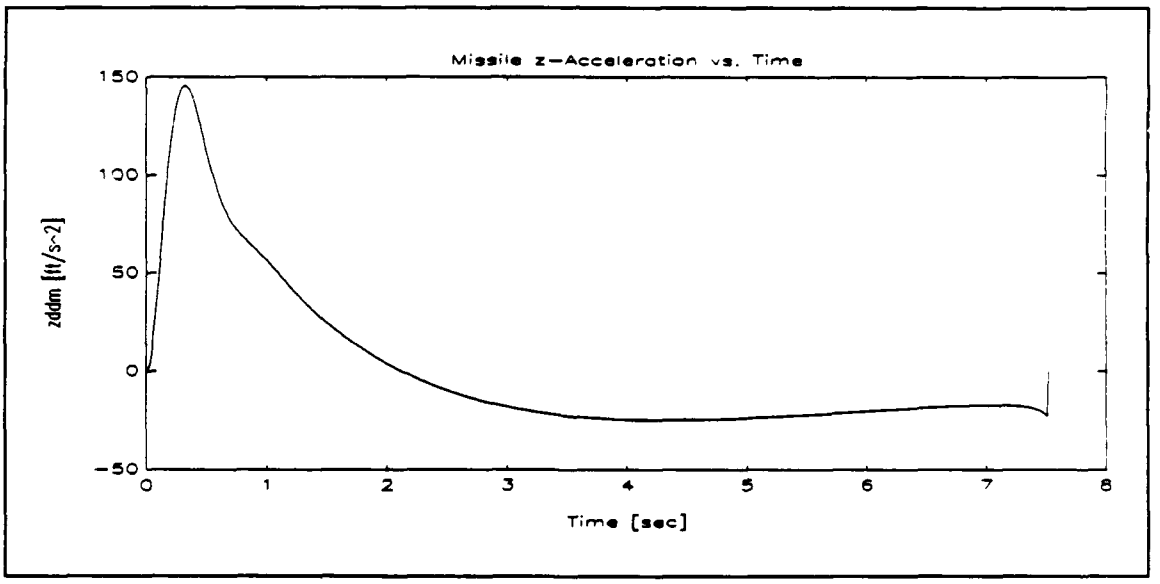


Figure 36. Proportional Navigation Scenario 1: Missile Acceleration (z-component) Time History

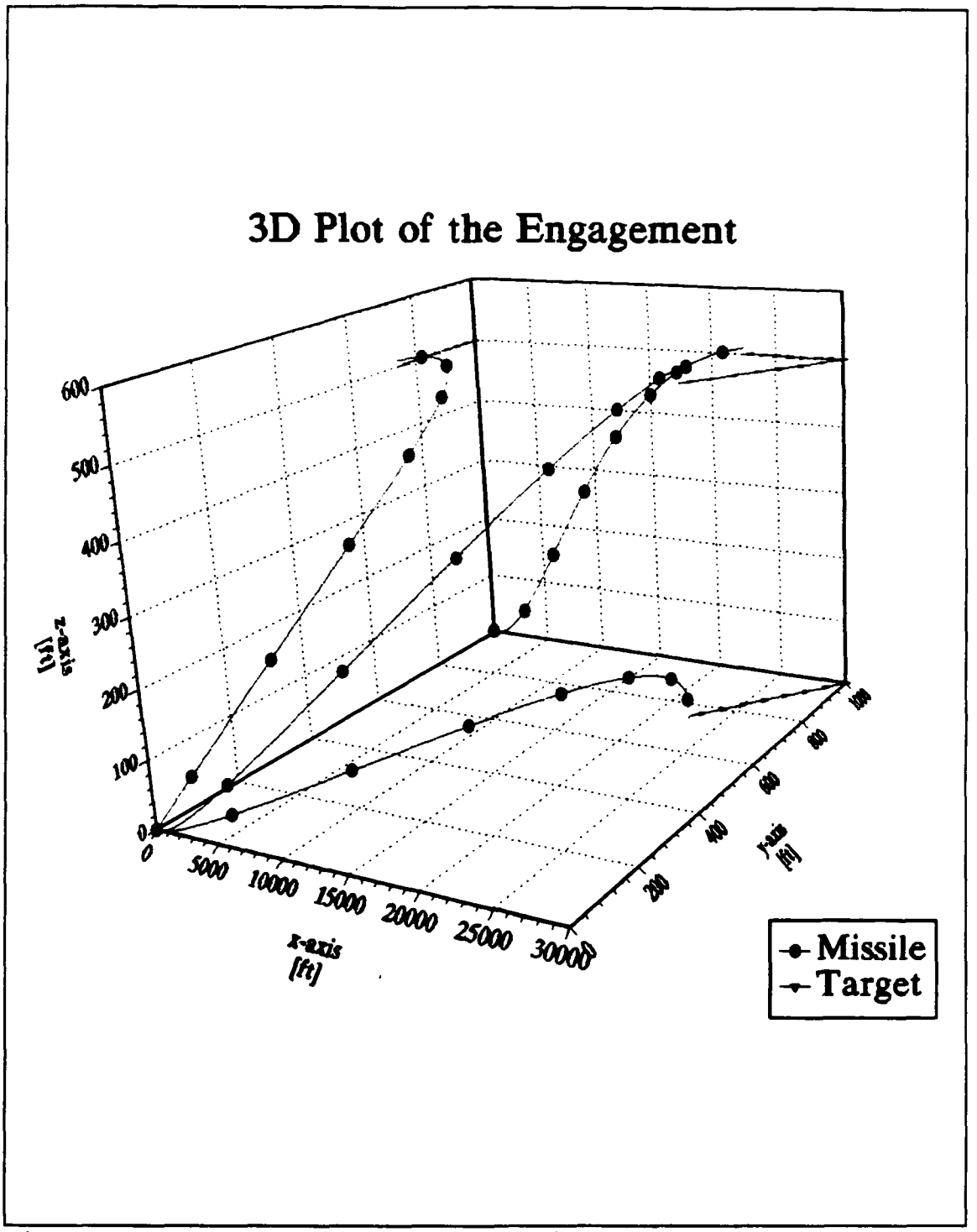


Figure 37. Proportional Navigation Scenario 1: Three-Dimensional Plot of the Engagement

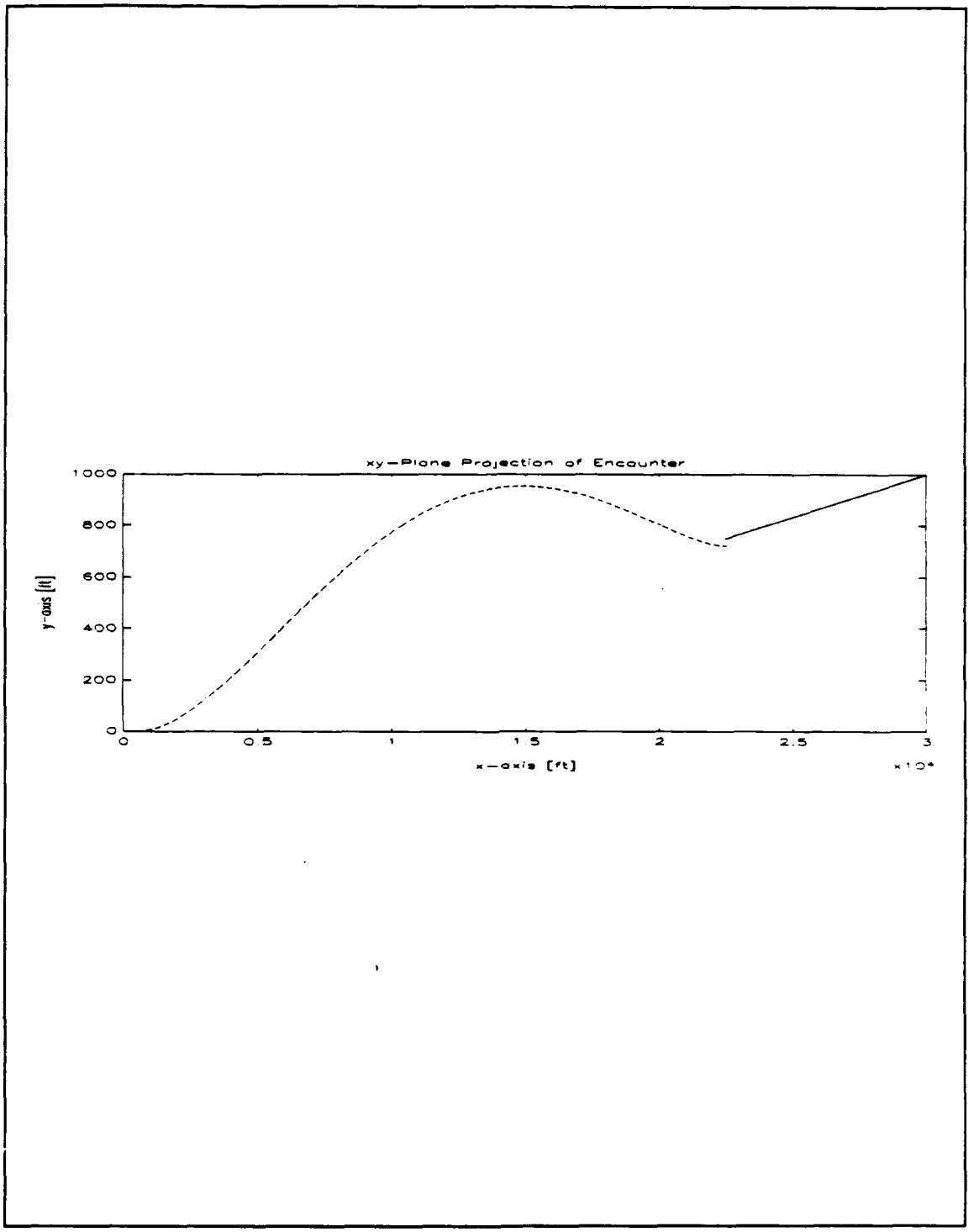


Figure 38. Proportional Navigation Scenario 1: Yaw Plane Projection of the Engagement

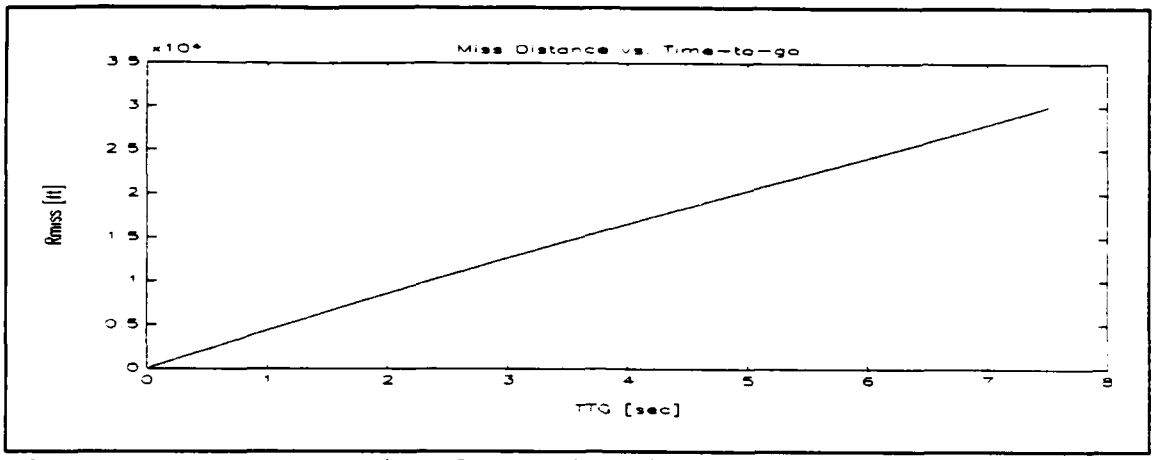


Figure 39. Proportional Navigation Scenario 2: Time-To-Go vs. Missile-to-Target Range

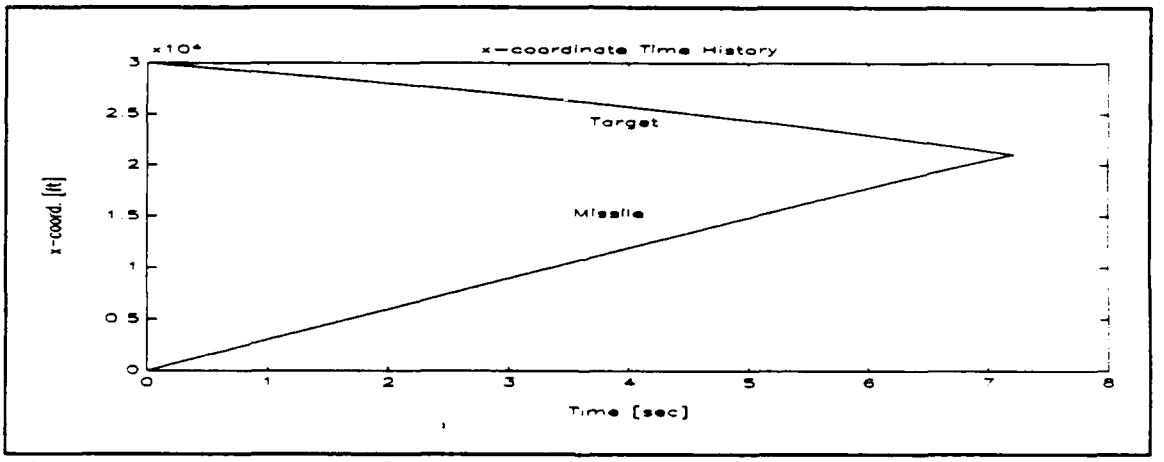


Figure 40. Proportional Navigation Scenario 2: Missile and Target x-coordinate Time History

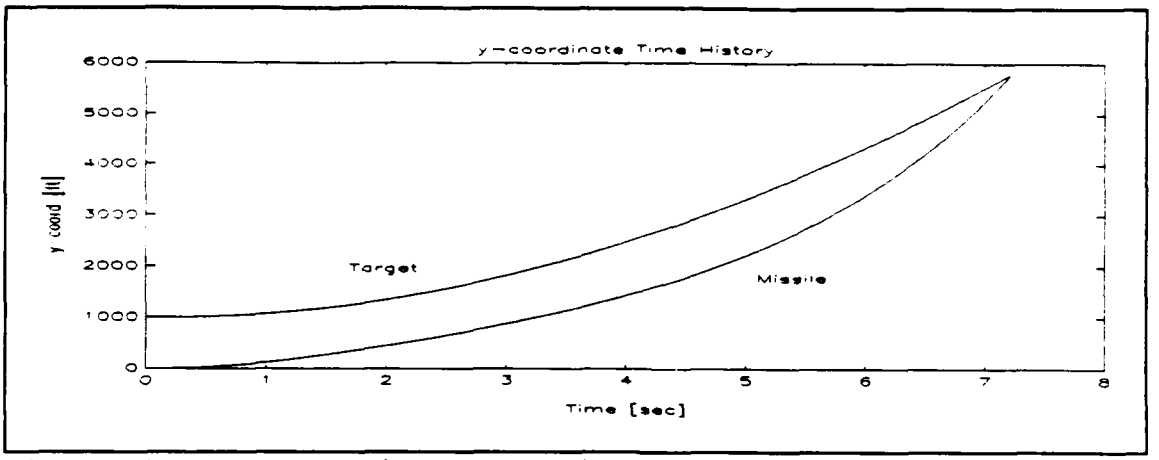


Figure 41. Proportional Navigation Scenario 2: Missile and Target y-coordinate Time History

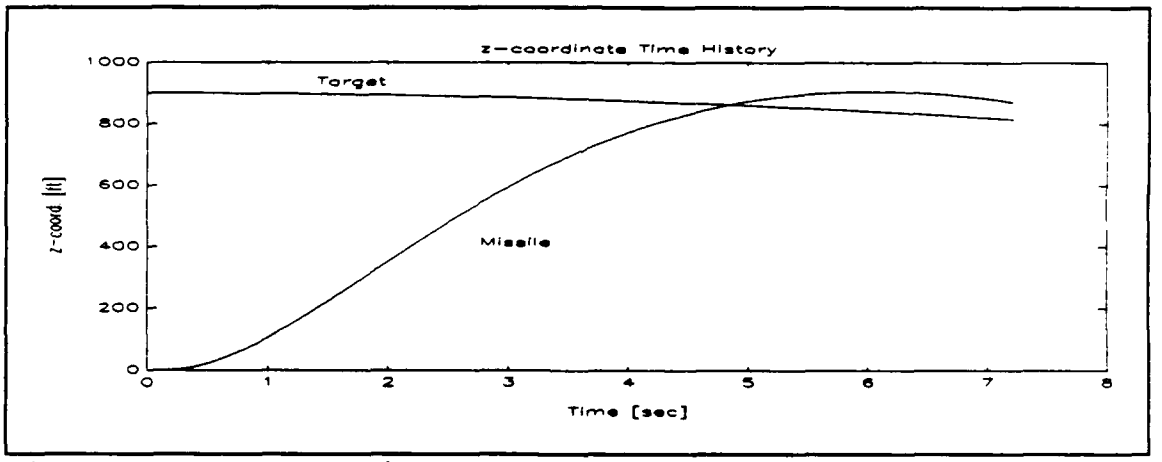


Figure 42. Proportional Navigation Scenario 2: Missile and Target z-coordinate Time History

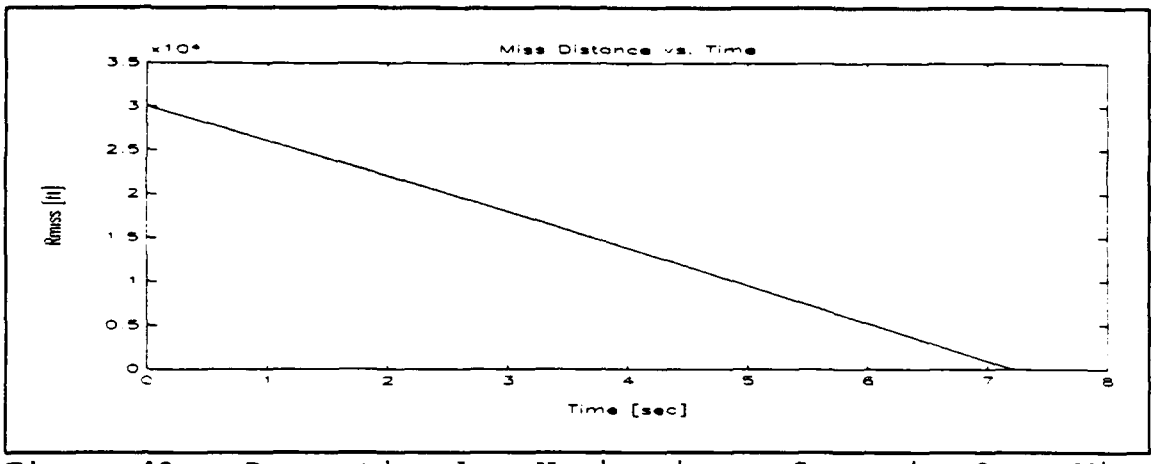


Figure 43. Proportional Navigation Scenario 2: Miss Distance Time History

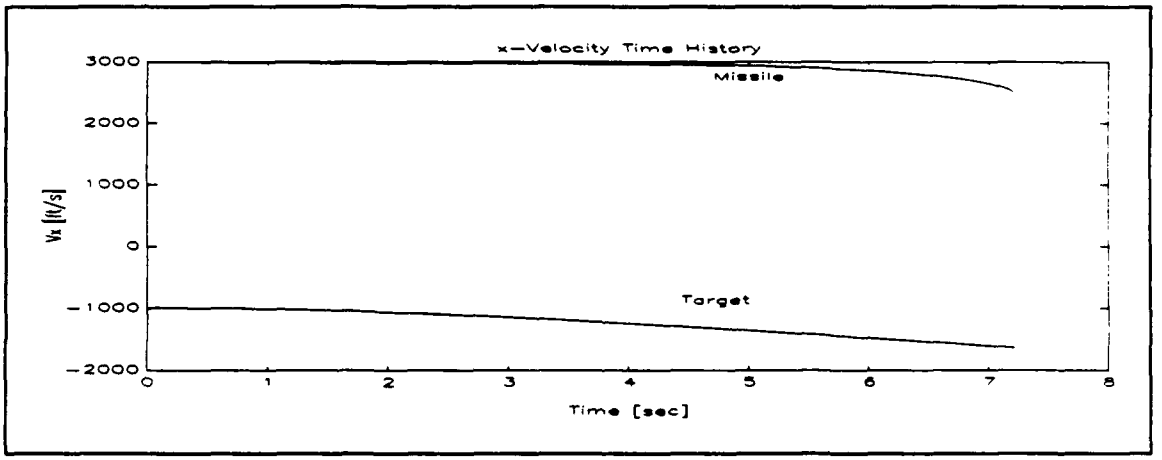


Figure 44. Proportional Navigation Scenario 2: Missile and Target Velocity (x-component) Time History

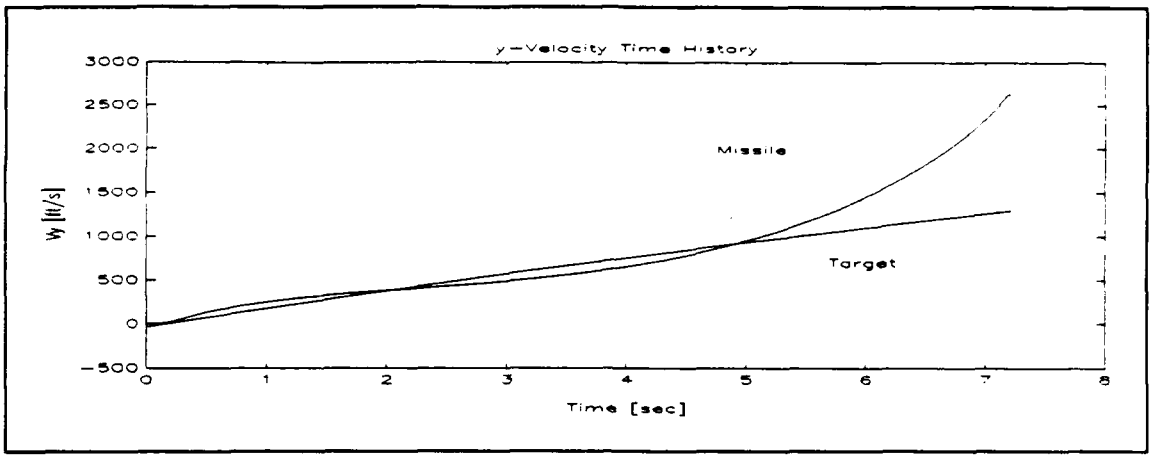


Figure 45. Proportional Navigation Scenario 2: Missile and Target Velocity (y-component) Time History

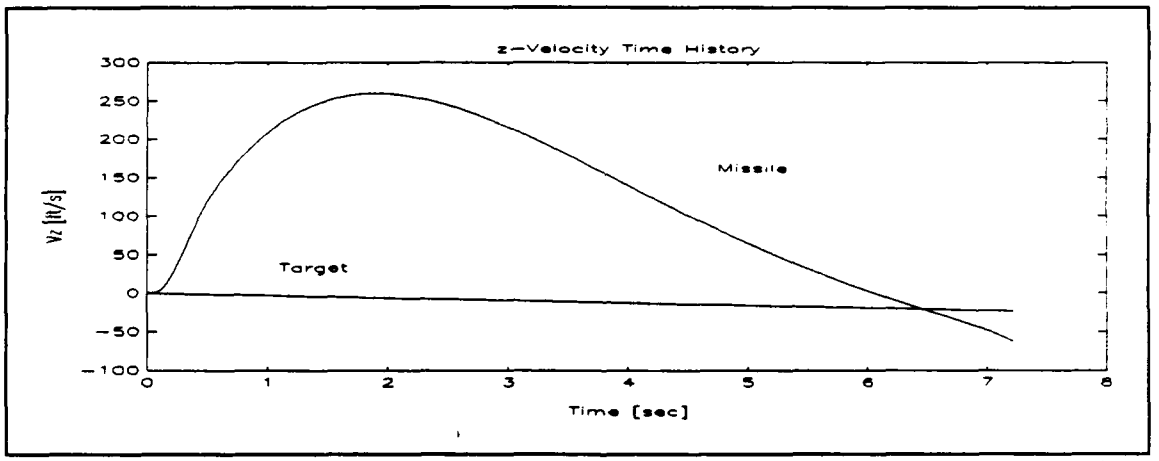


Figure 46. Proportional Navigation Scenario 2: Missile and Target Velocity (z-component) Time History

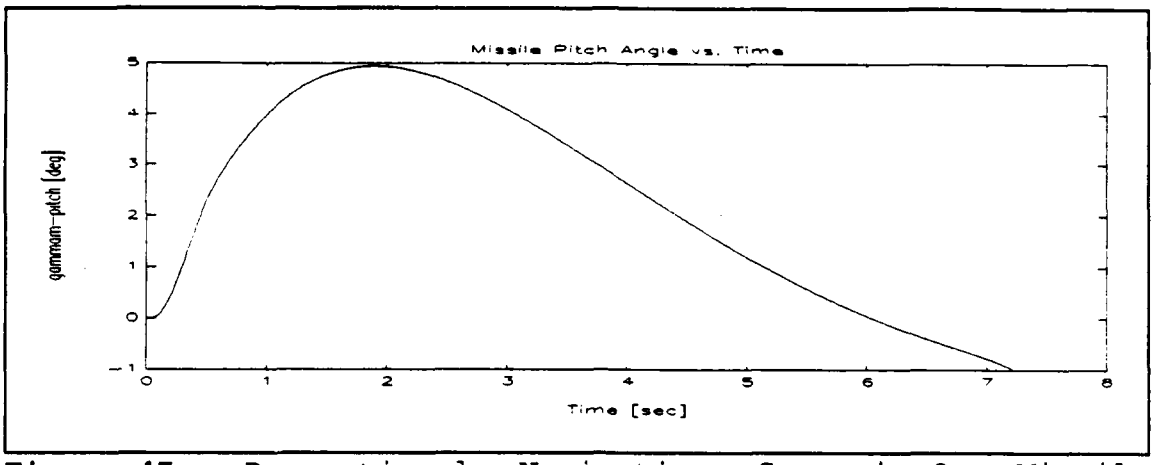


Figure 47. Proportional Navigation Scenario 2: Missile Pitch Angle Time History

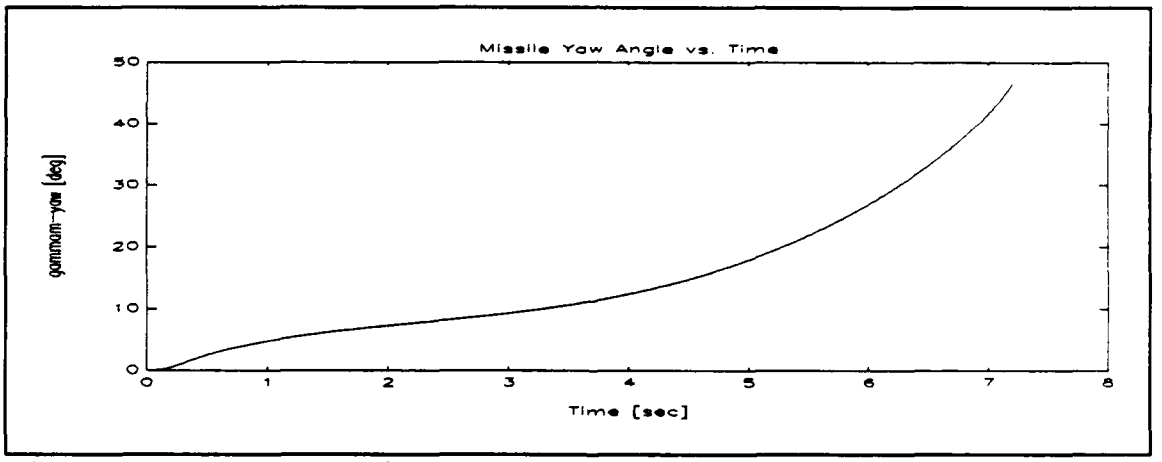


Figure 48. Proportional Navigation Scenario 2: Missile Yaw Angle Time History

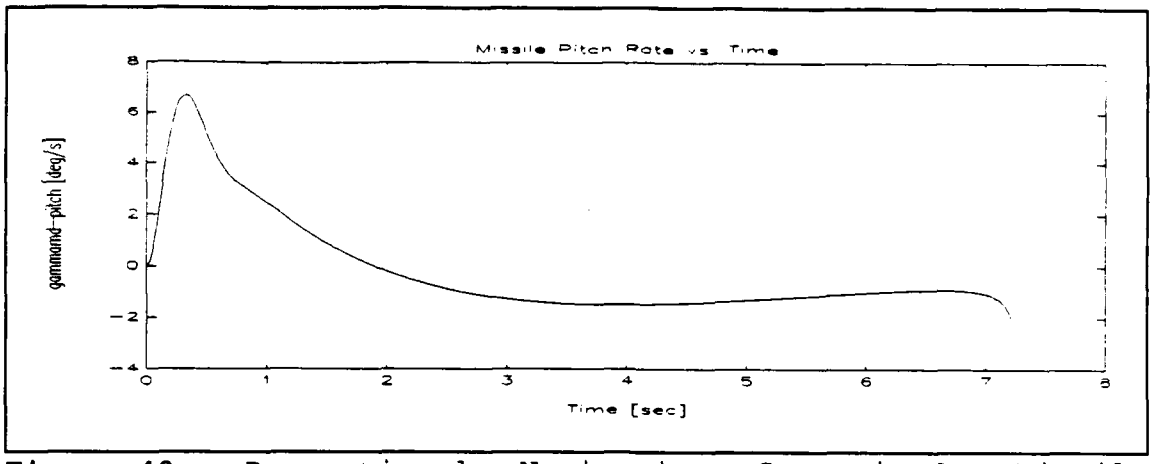


Figure 49. Proportional Navigation Scenario 2: Missile Pitch Rate Time History

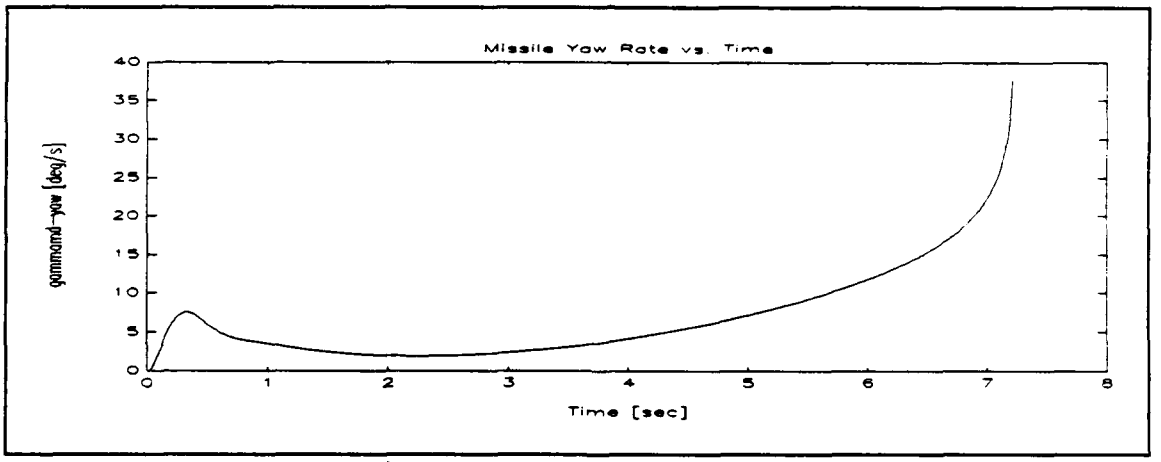


Figure 50. Proportional Navigation Scenario 2: Missile Yaw Rate Time History

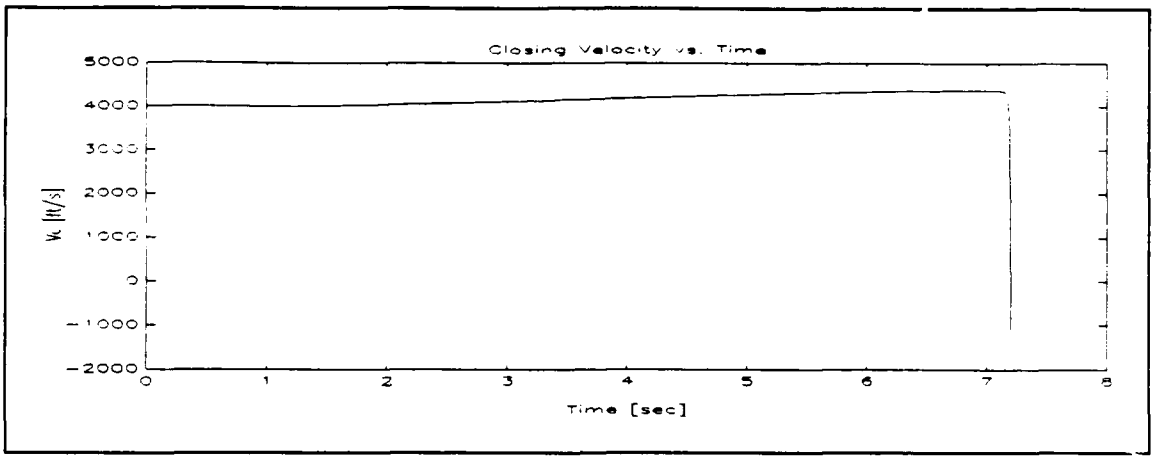


Figure 51. Proportional Navigation Scenario 2: Closing Velocity Time History

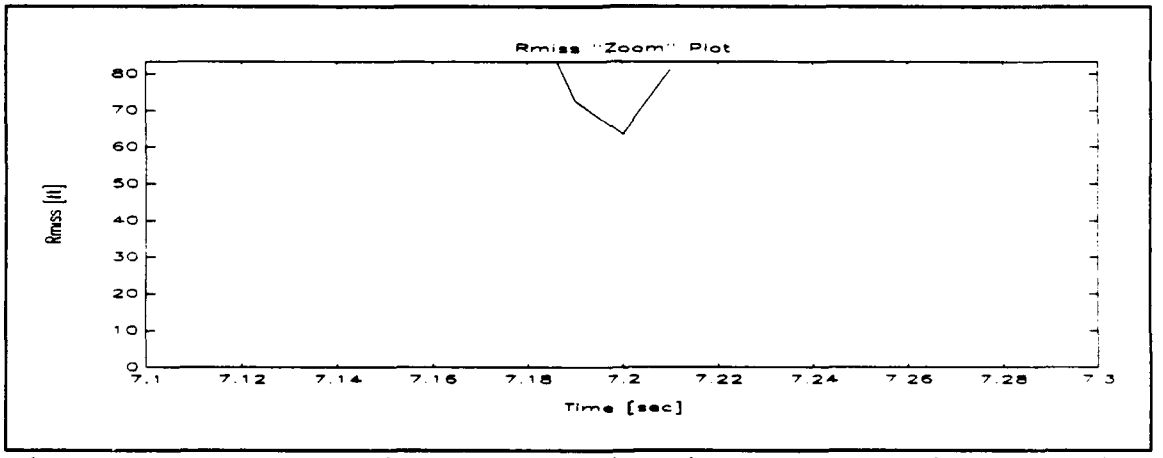


Figure 52. Proportional Navigation Scenario 2: Miss Distance and Time-of-Closest-Point-of Approach Enhancement Plot

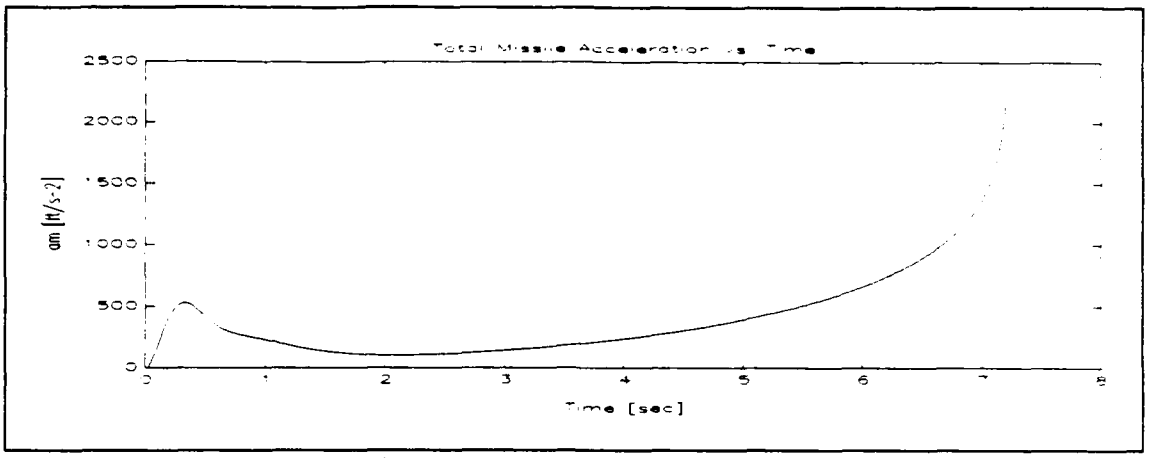


Figure 53. Proportional Navigation Scenario 2: Missile Total Acceleration Time History

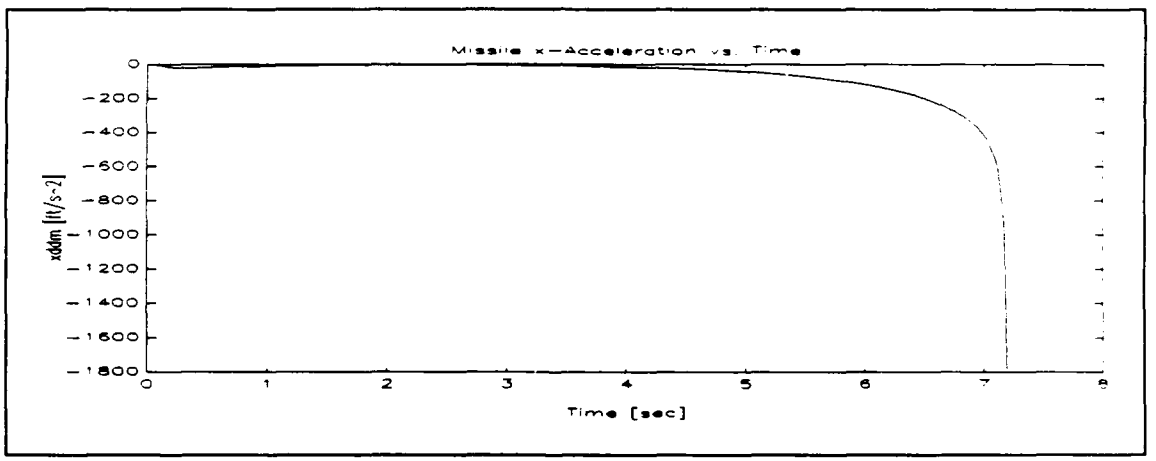


Figure 54. Proportional Navigation Scenario 2: Missile Acceleration (x-component) Time History

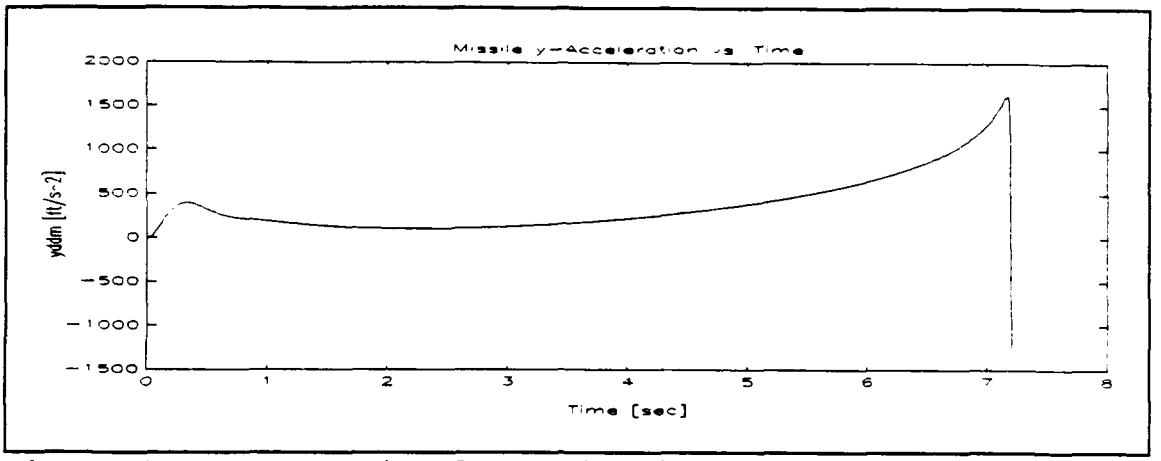


Figure 55. Proportional Navigation Scenario 2: Missile Acceleration (y-component) Time History

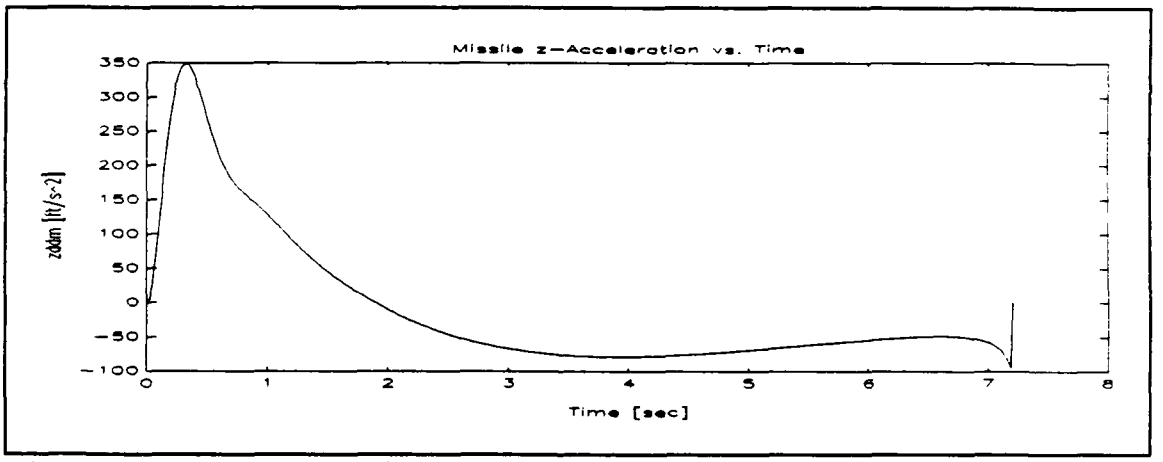


Figure 56. Proportional Navigation Scenario 2: Missile Acceleration (z-component) Time History

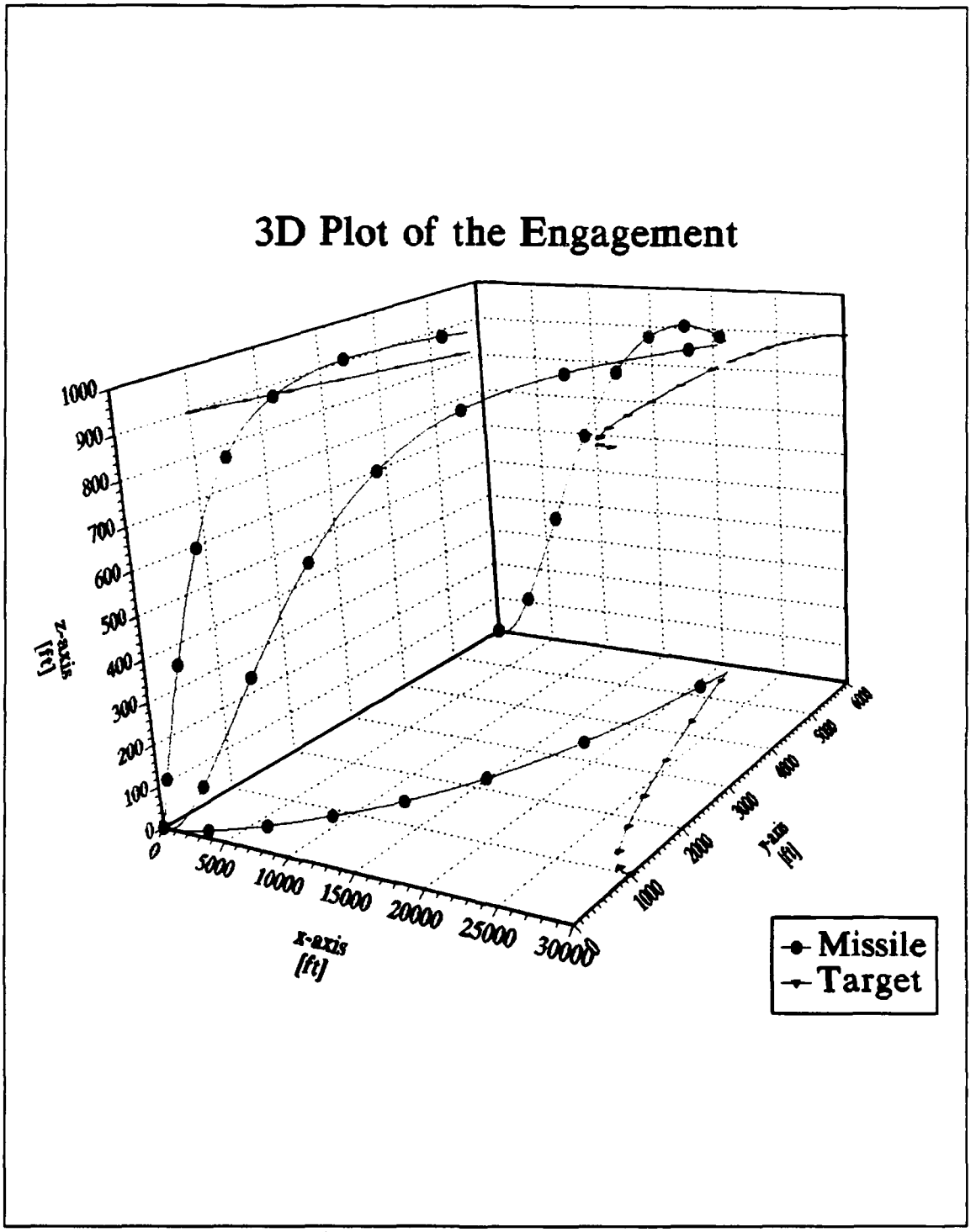


Figure 57. Proportional Navigation Scenario 2: Three-Dimensional Plot of the Engagement

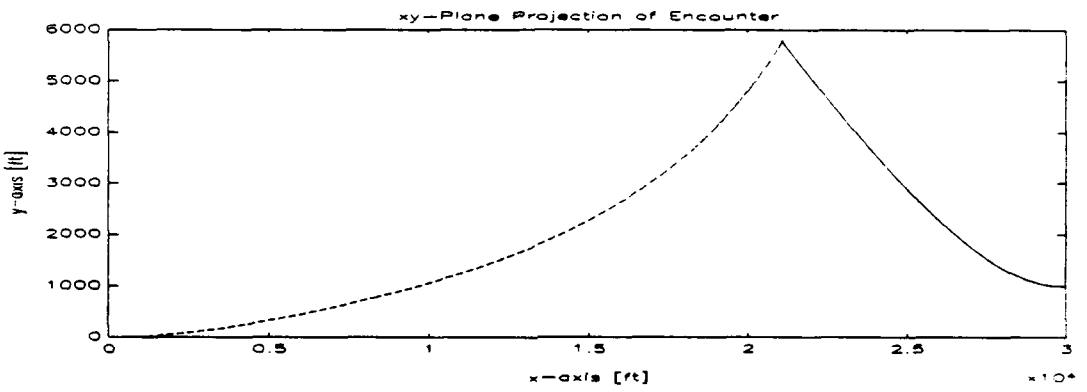


Figure 58. Proportional Navigation Scenario 2: Yaw Plane Projection of the Engagement

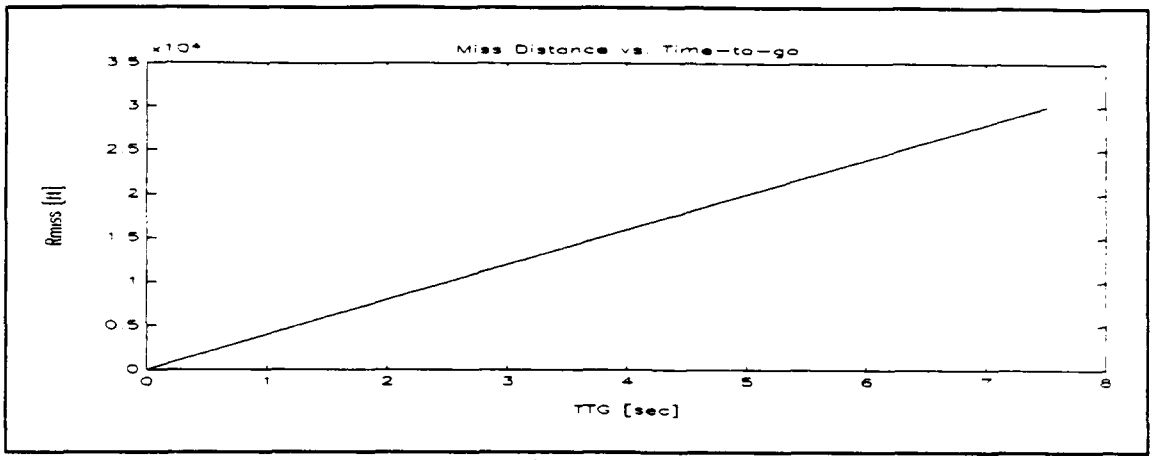


Figure 59. CLOS Scenario 1: Time-To-Go vs Missile-to-Target Range

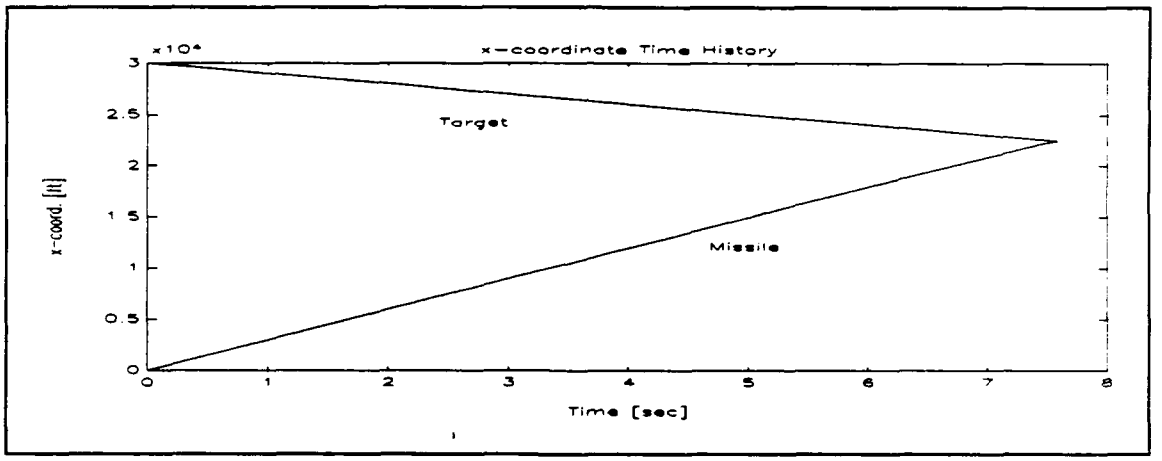


Figure 60. CLOS Scenario 1: Missile and Target x-coordinate Time History

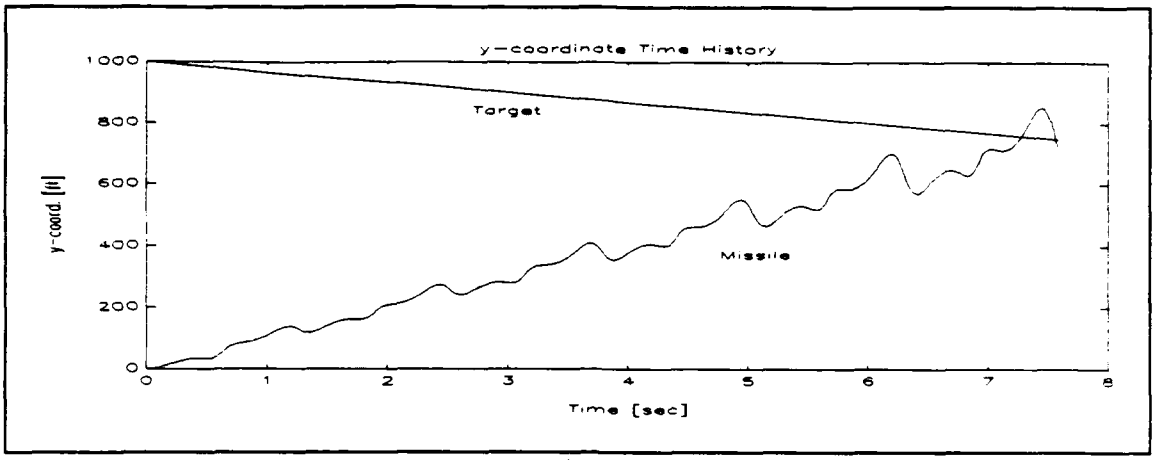


Figure 61. CLOS Scenario 1: Missile and Target y-coordinate Time History

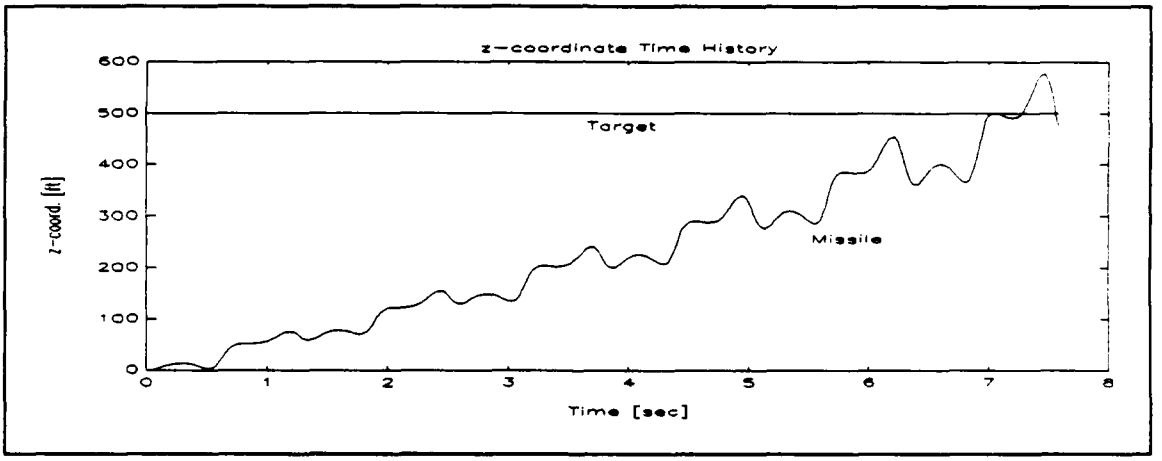


Figure 62. CLOS Scenario 1: Missile and Target z-coordinate Time History

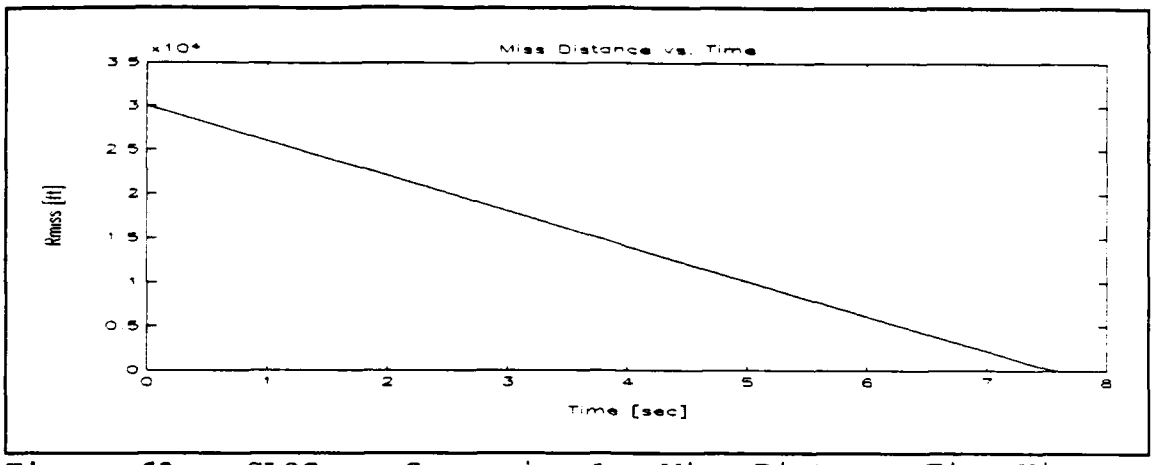


Figure 63. CLOS Scenario 1: Miss Distance Time History

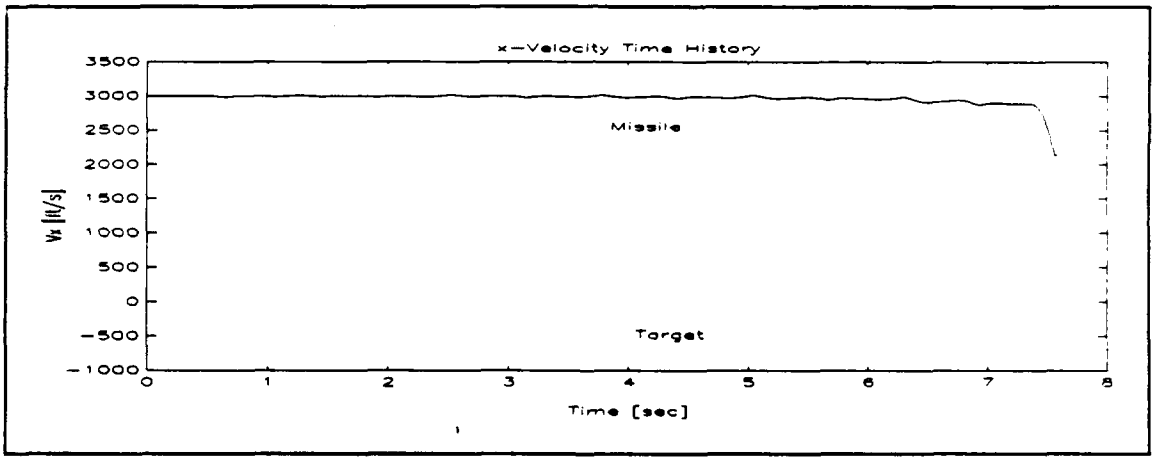


Figure 64. CLOS Scenario 1: Missile and Target Velocity (x-component) Time History

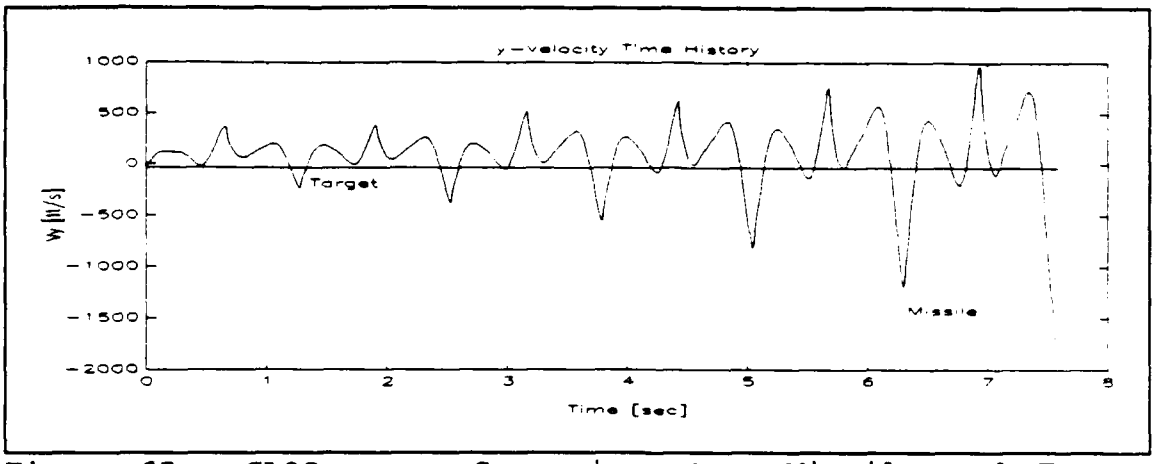


Figure 65. CLOS Scenario 1: Missile and Target Velocity (y-component) Time History

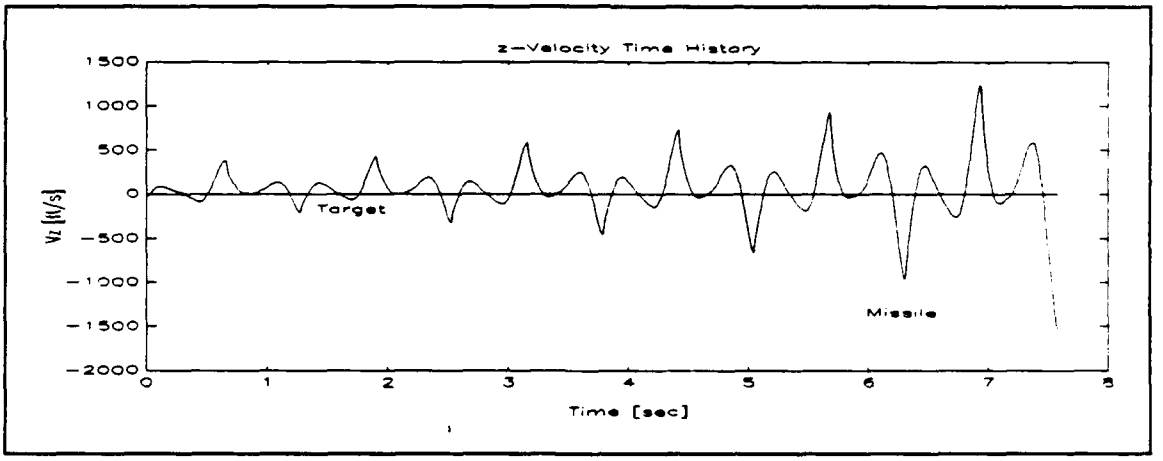


Figure 66. CLOS Scenario 1: Missile and Target Velocity (z-component) Time History

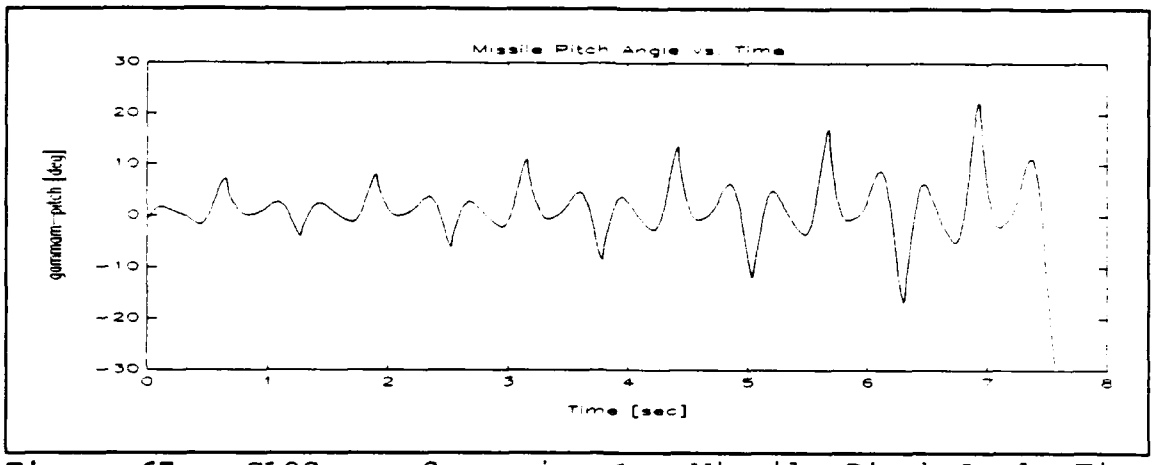


Figure 67. CLOS Scenario 1: Missile Pitch Angle Time History

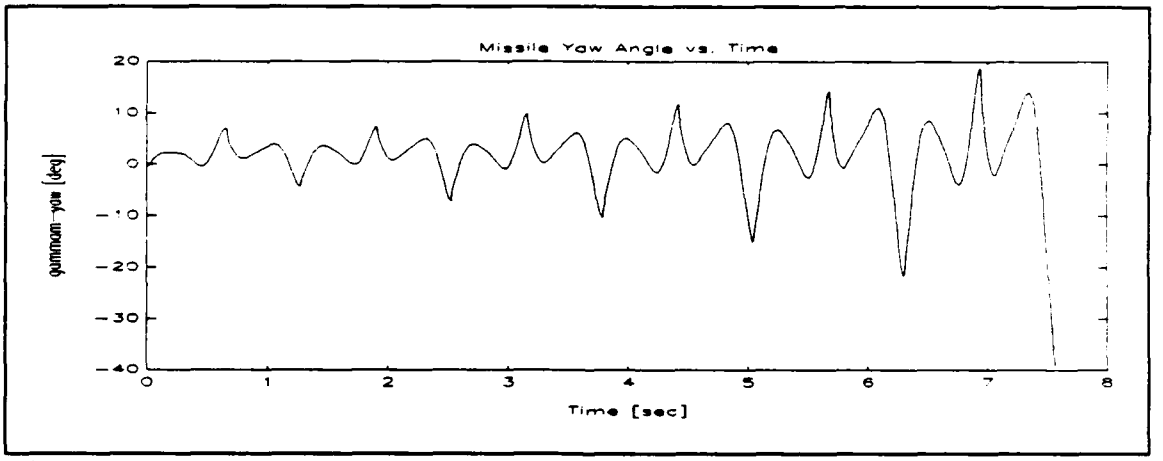


Figure 68. CLOS Scenario 1: Missile Yaw Angle Time History

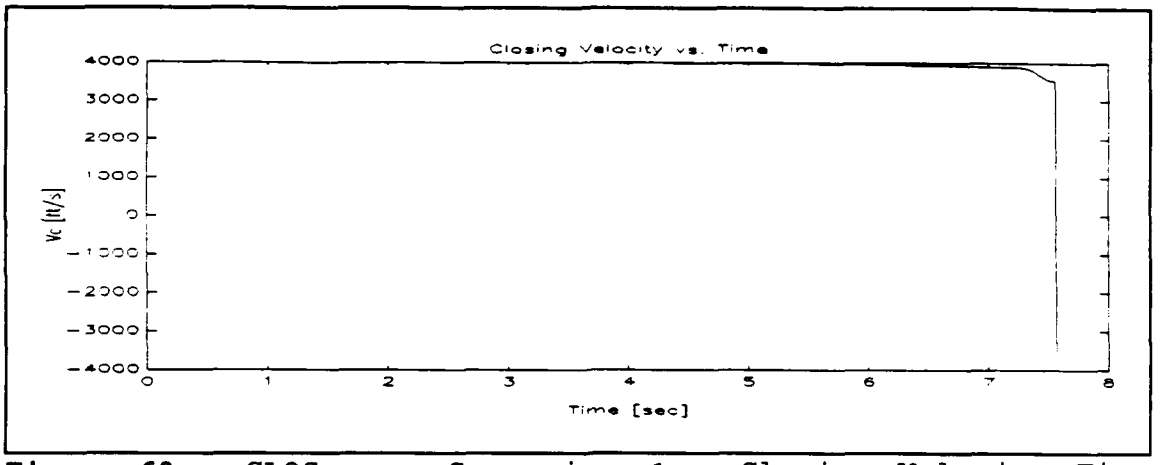


Figure 69. CLOS Scenario 1: Closing Velocity Time History

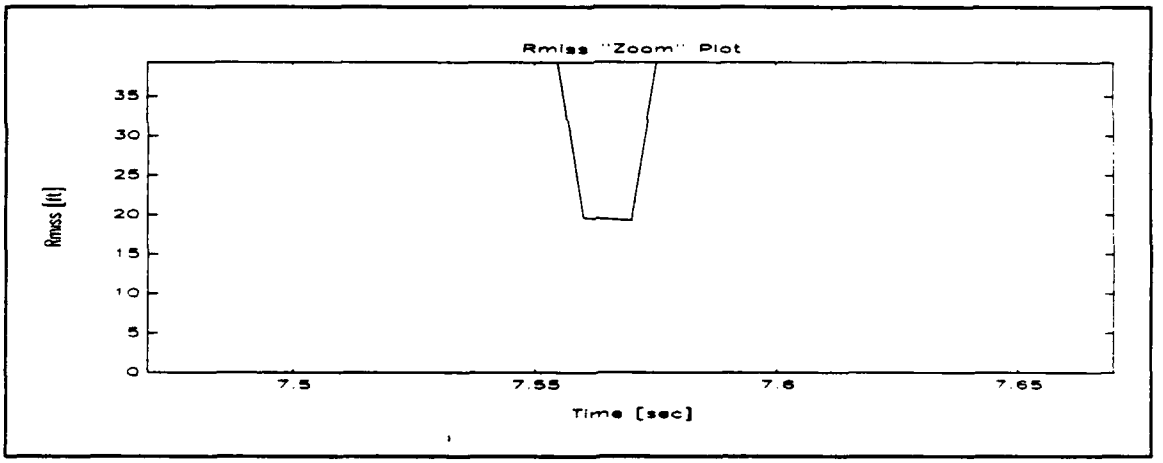


Figure 70. CLOS Scenario 1: Miss Distance and Time-of-Closest-Point-of Approach Enhancement Plot

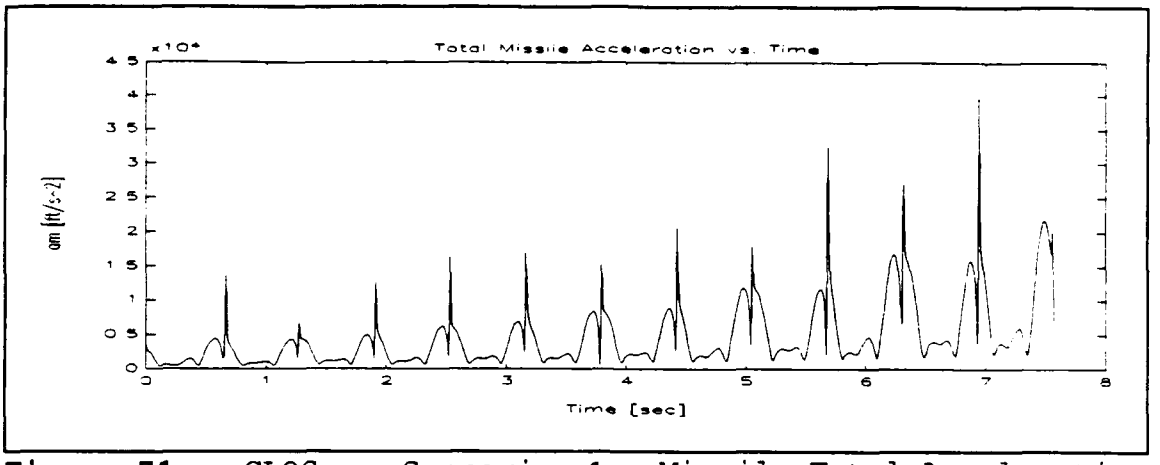


Figure 71. CLOS Scenario 1: Missile Total Acceleration Time History

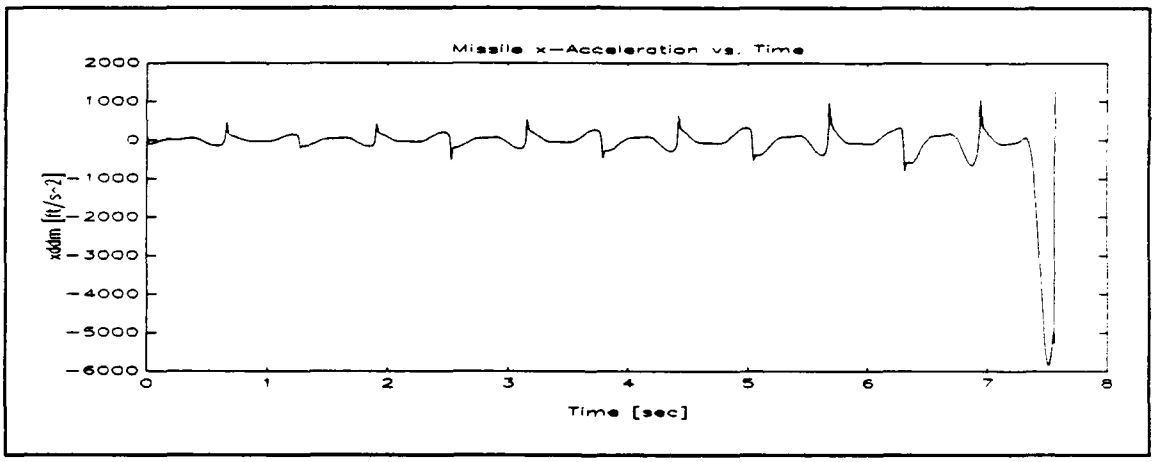


Figure 72. CLOS Scenario 1: Missile Acceleration (x-component) Time History

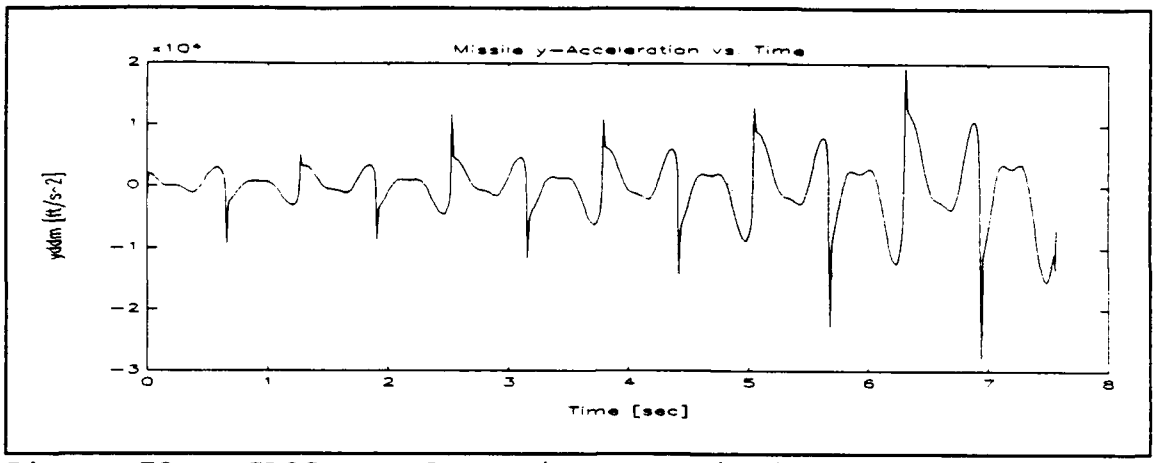


Figure 73. CLOS Scenario 1: Missile Acceleration (y-component) Time History

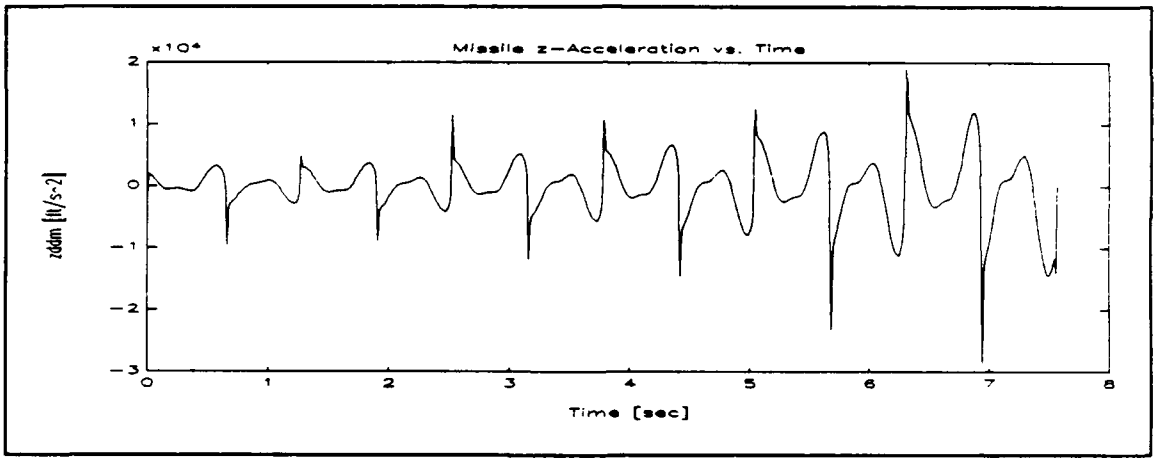


Figure 74. CLOS Scenario 1: Missile Acceleration (z-component) Time History

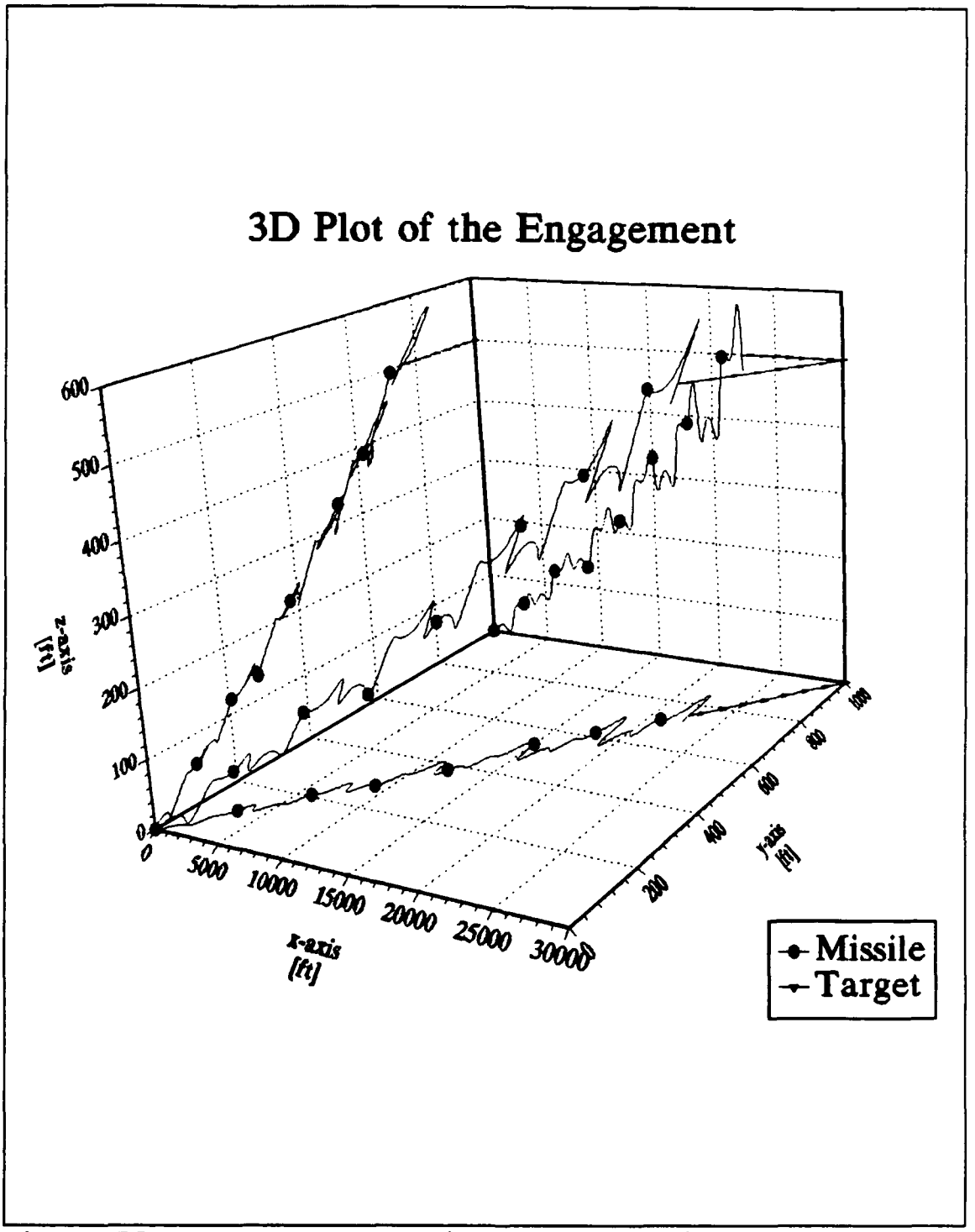


Figure 75. CLOS Scenario 1: Three-Dimensional Plot of the Engagement

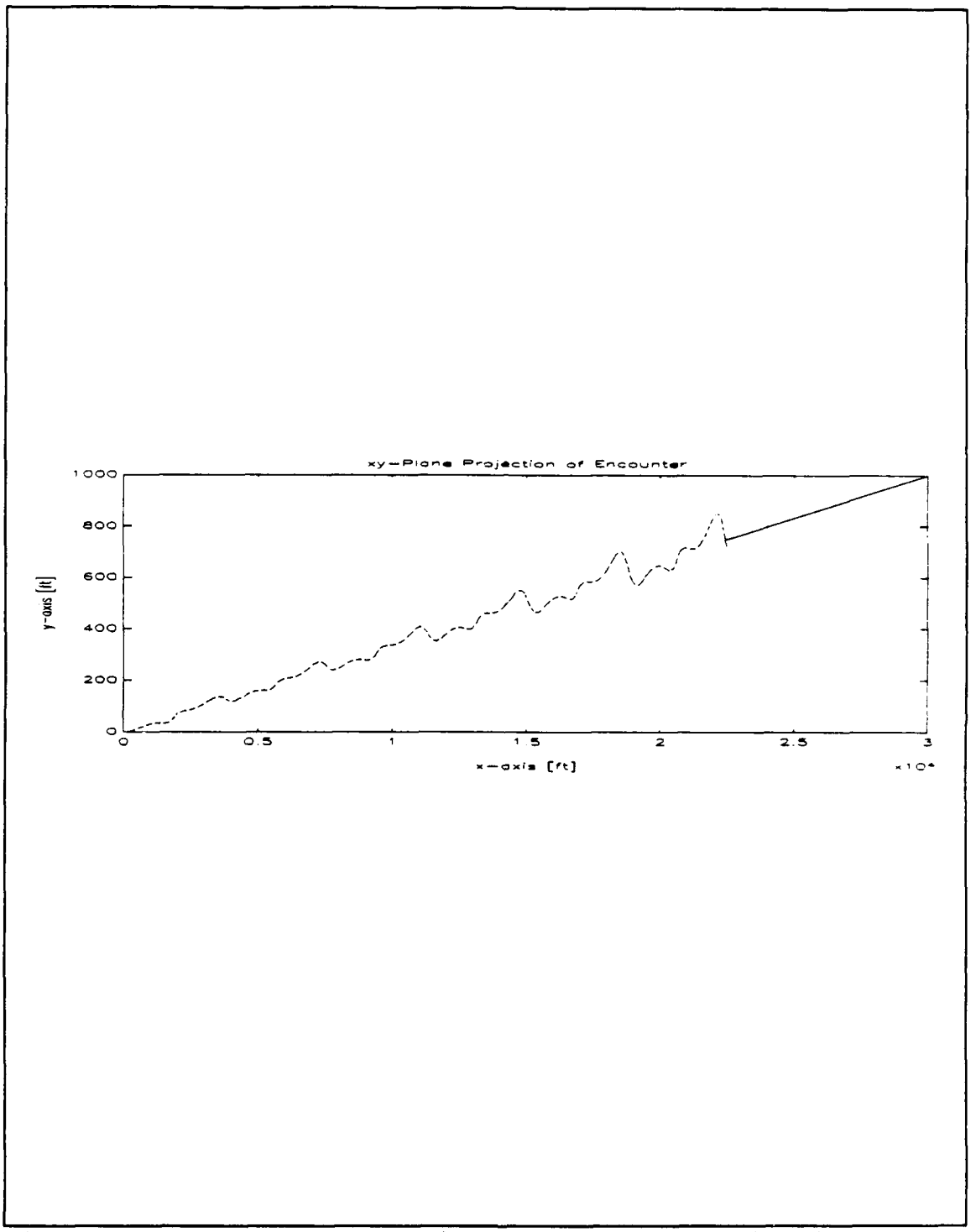


Figure 76. CLOS Scenario 1: Yaw Plane Projection of the Engagement

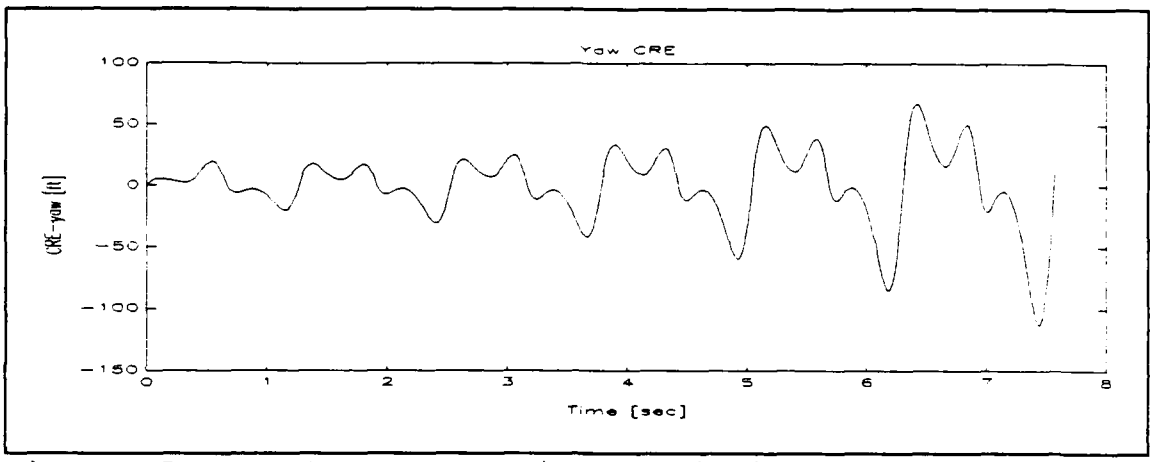


Figure 77. CLOS Scenario 1: Missile Cross Range Error (Yaw component) Time History

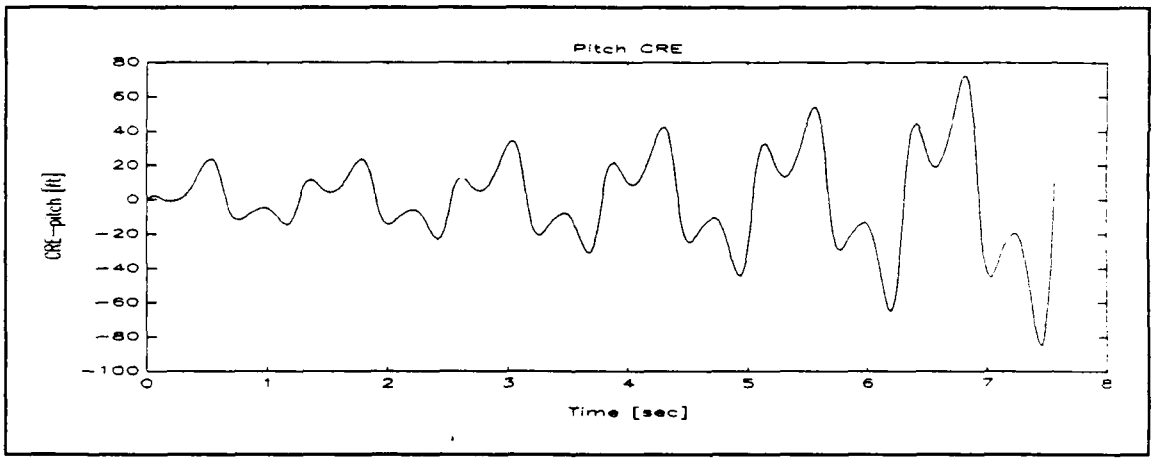


Figure 78. CLOS Scenario 1: Missile Cross Range Error (Pitch component) Time History

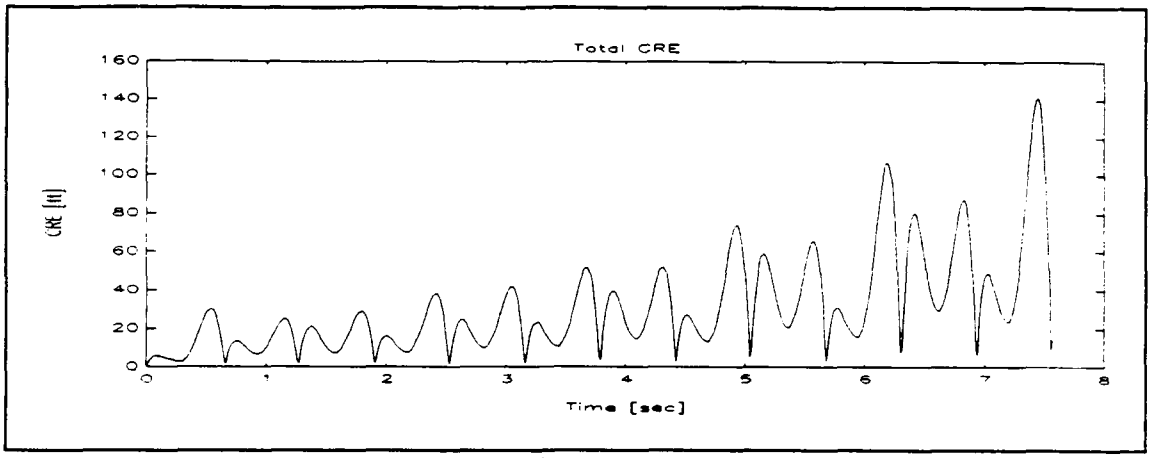


Figure 79. CLOS Scenario 1: Total Missile Cross Range Error Time History

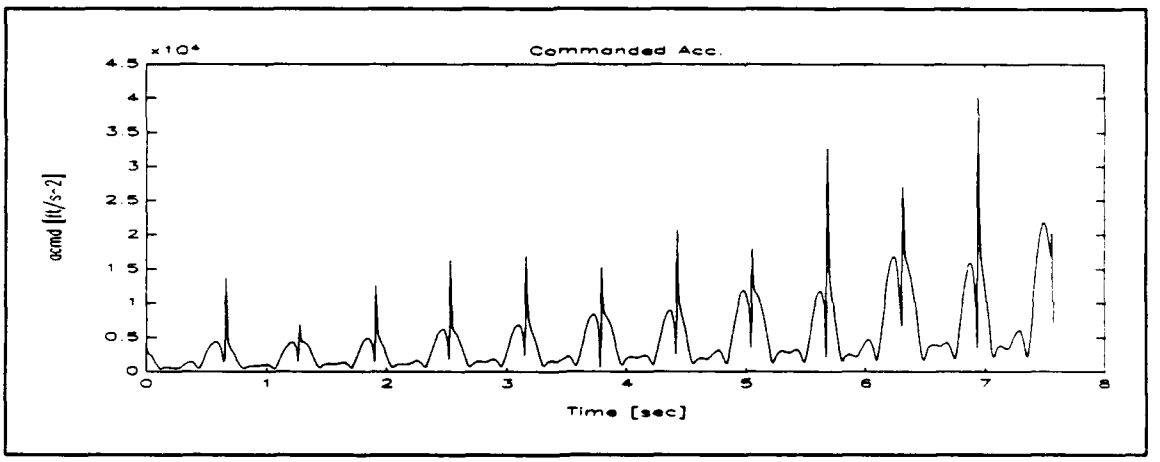


Figure 80. CLOS Scenario 1: Total Commanded Missile Acceleration Time History

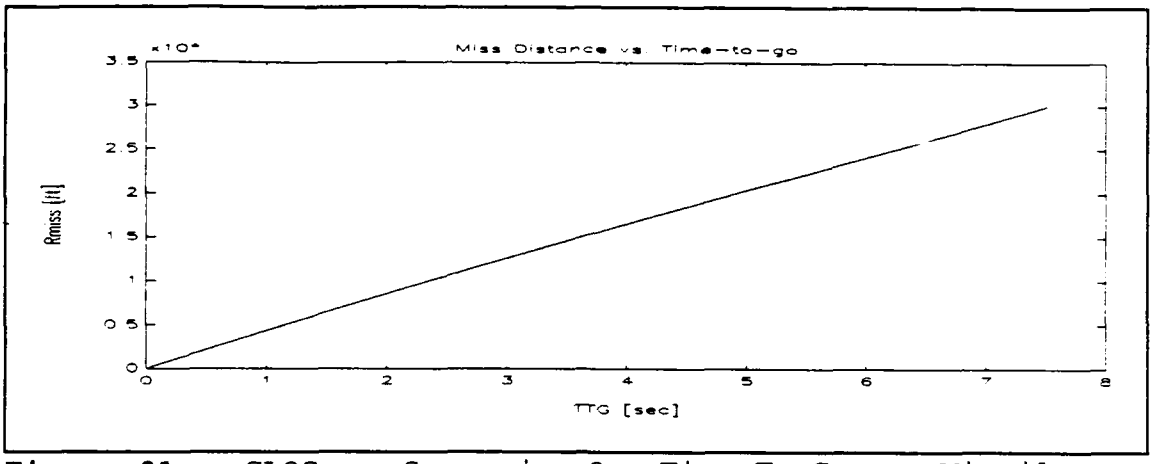


Figure 81. CLOS Scenario 2: Time-To-Go vs. Missile-to-Target Range

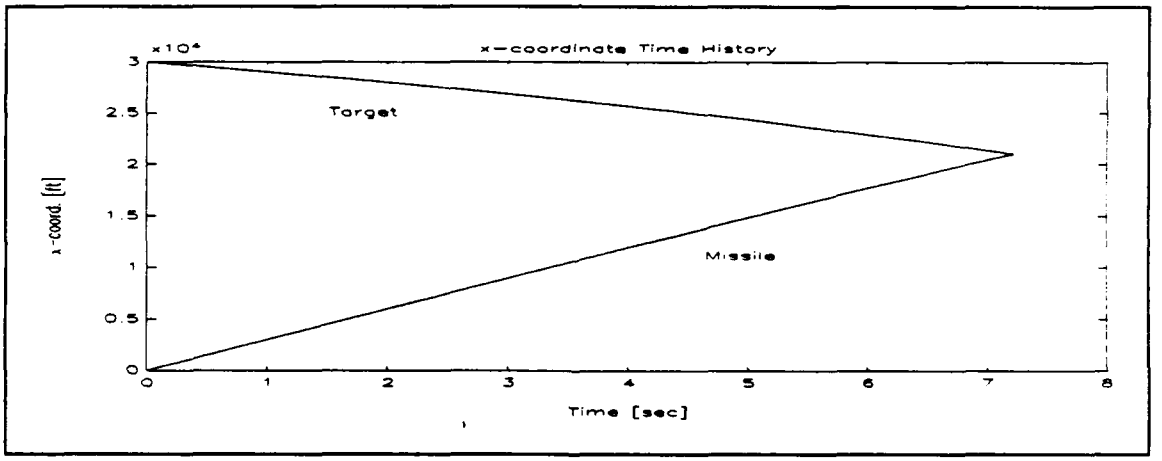


Figure 82. CLOS Scenario 2: Missile and Target x-coordinate Time History

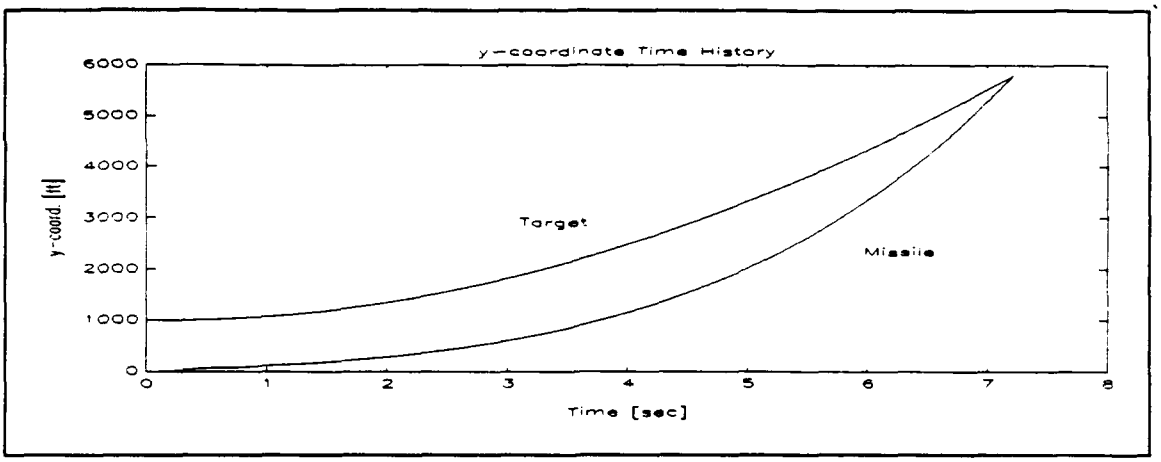


Figure 83. CLOS Scenario 2: Missile and Target y-coordinate Time History

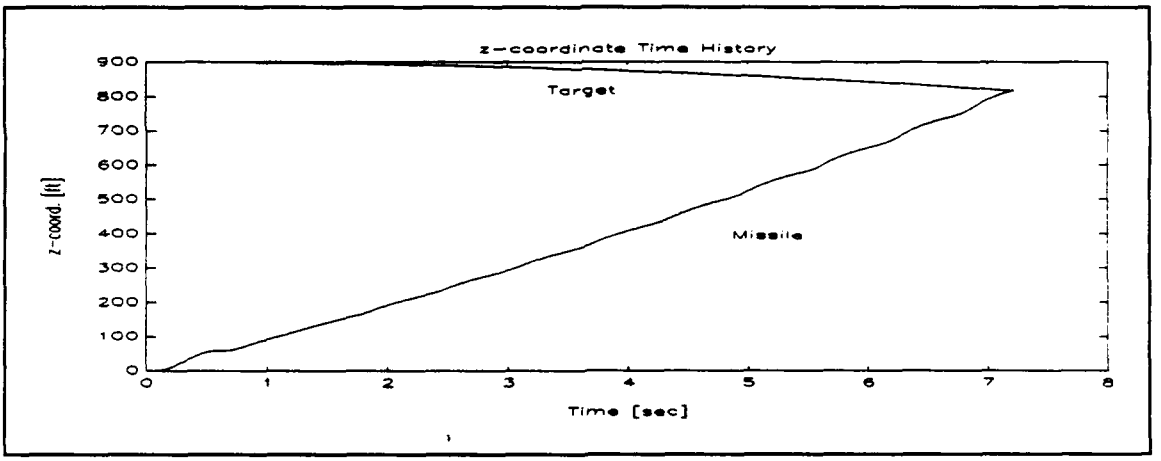


Figure 84. CLOS Scenario 2: Missile and Target z-coordinate Time History

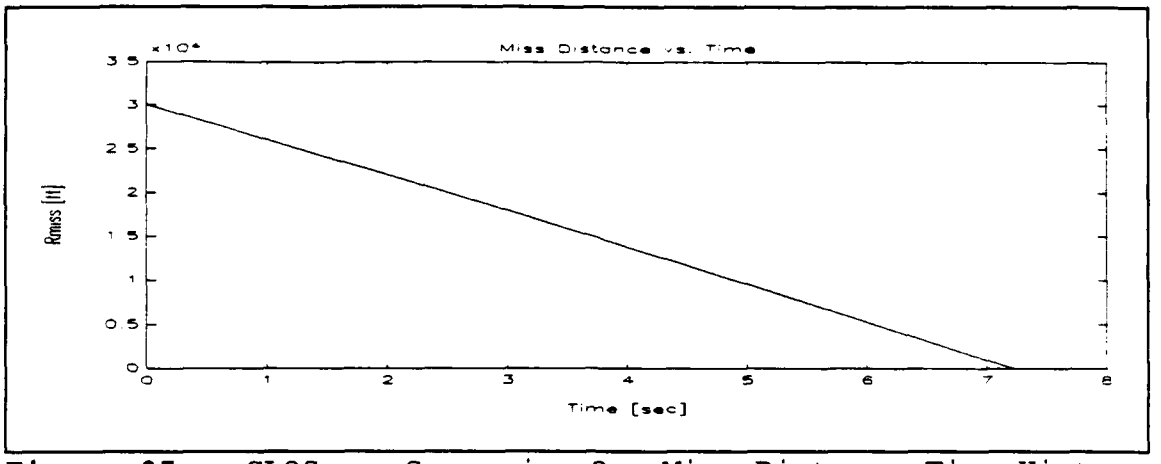


Figure 85. CLOS Scenario 2: Miss Distance Time History

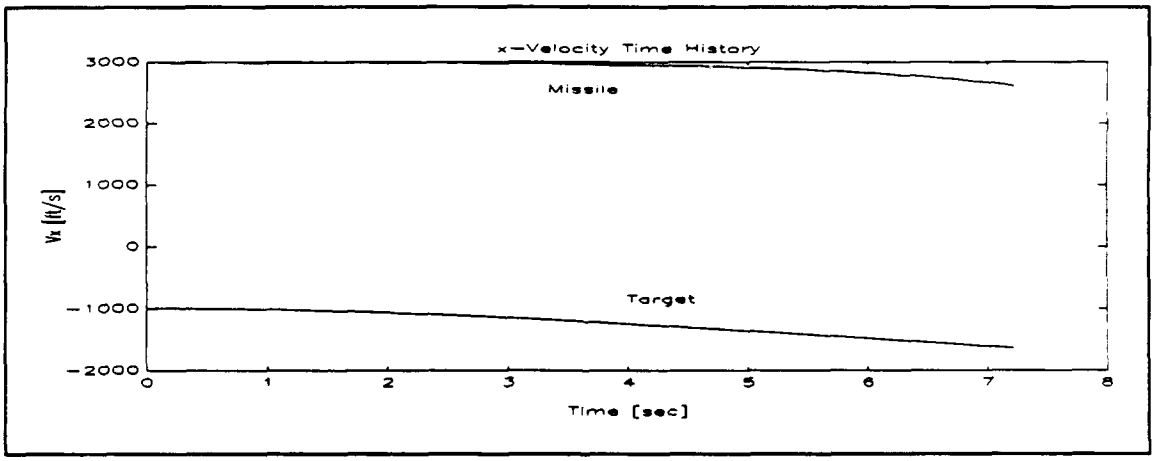


Figure 86. CLOS Scenario 2: Missile and Target Velocity (x-component) Time History

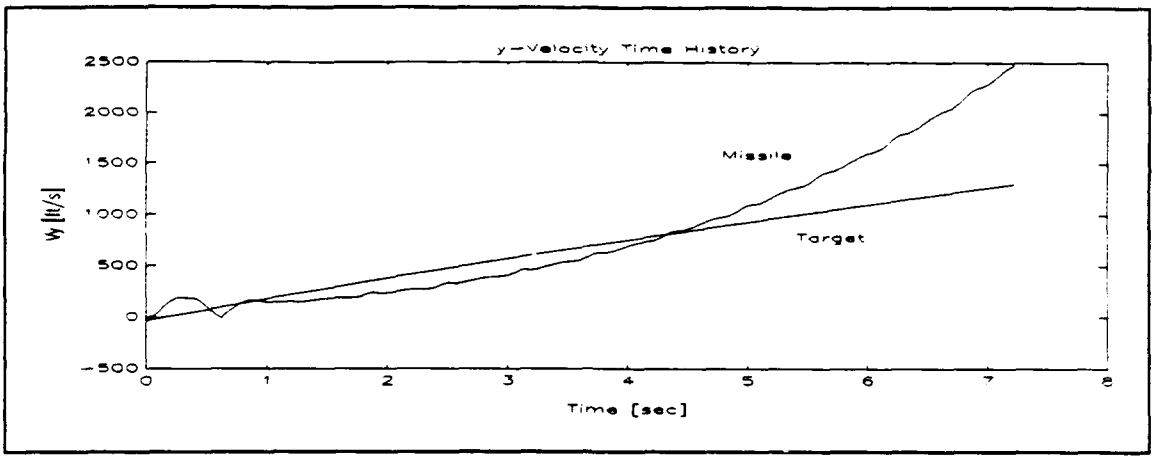


Figure 87. CLOS Scenario 2: Missile and Target Velocity (y-component) Time History

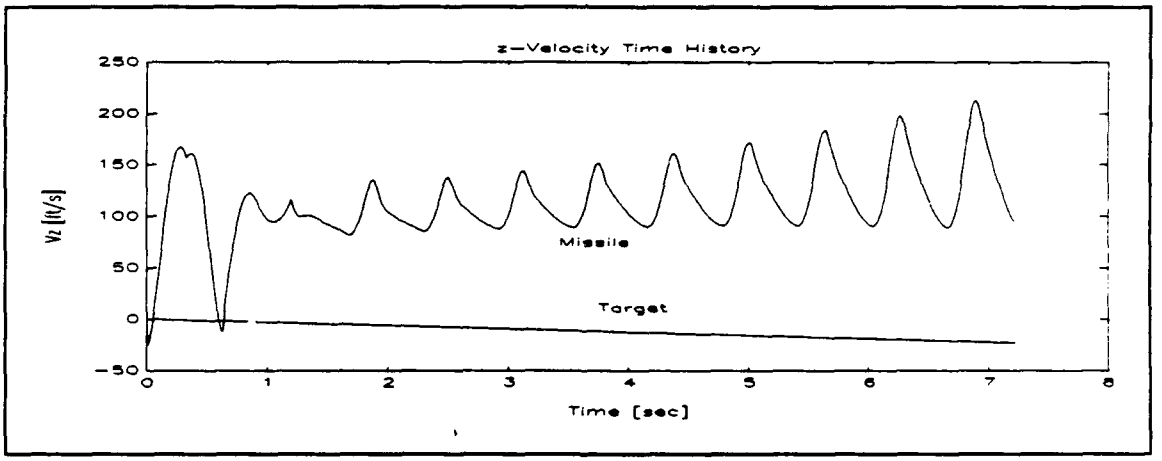


Figure 88. CLOS Scenario 2: Missile and Target Velocity (z-component) Time History

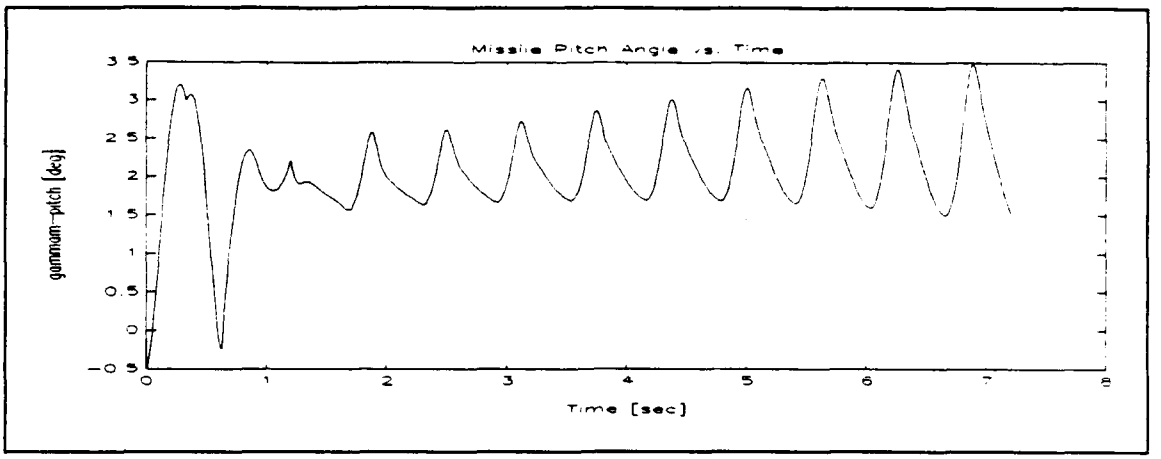


Figure 89. CLOS Scenario 2: Missile Pitch Angle Time History

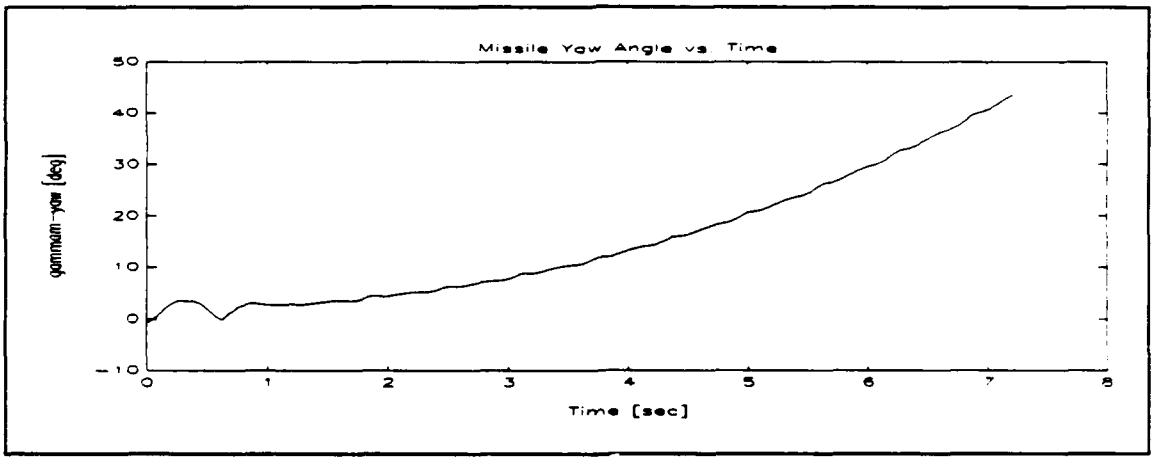


Figure 90. CLOS Scenario 2: Missile Yaw Angle Time History

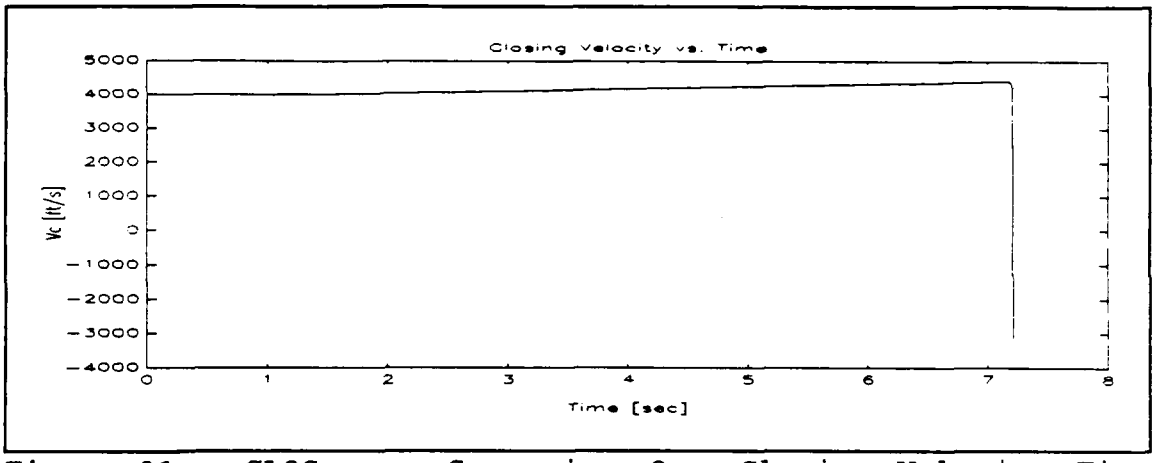


Figure 91. CLOS Scenario 2: Closing Velocity Time History

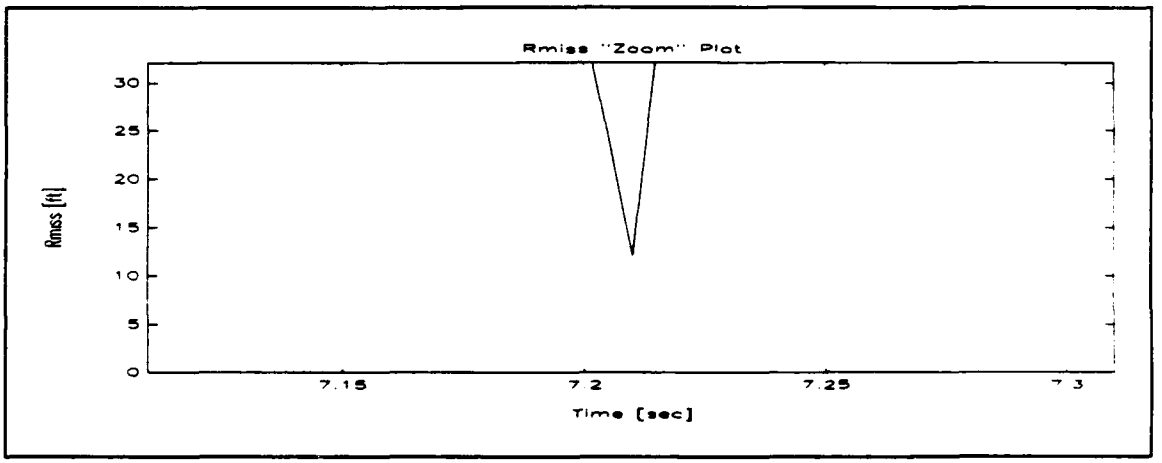


Figure 92. CLOS Scenario 2: Miss Distance and Time-of-Closest-Point-of Approach Enhancement Plot

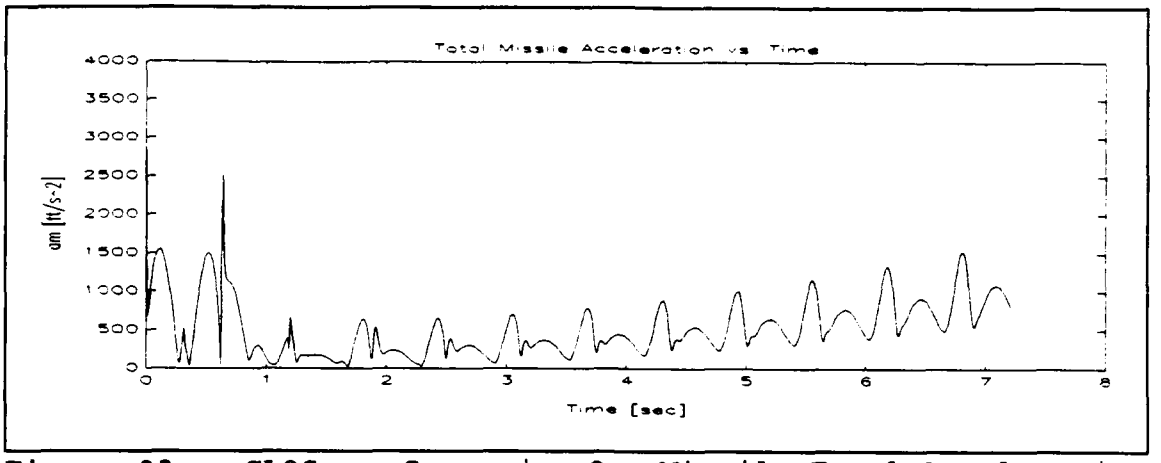


Figure 93. CLOS Scenario 2: Missile Total Acceleration Time History

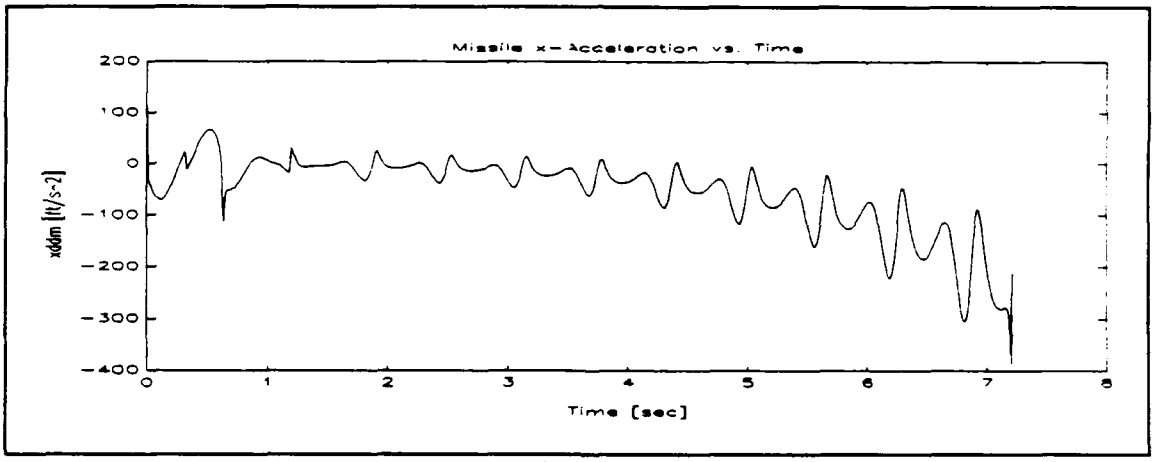


Figure 94. CLOS Scenario 2: Missile Acceleration (x-component) Time History

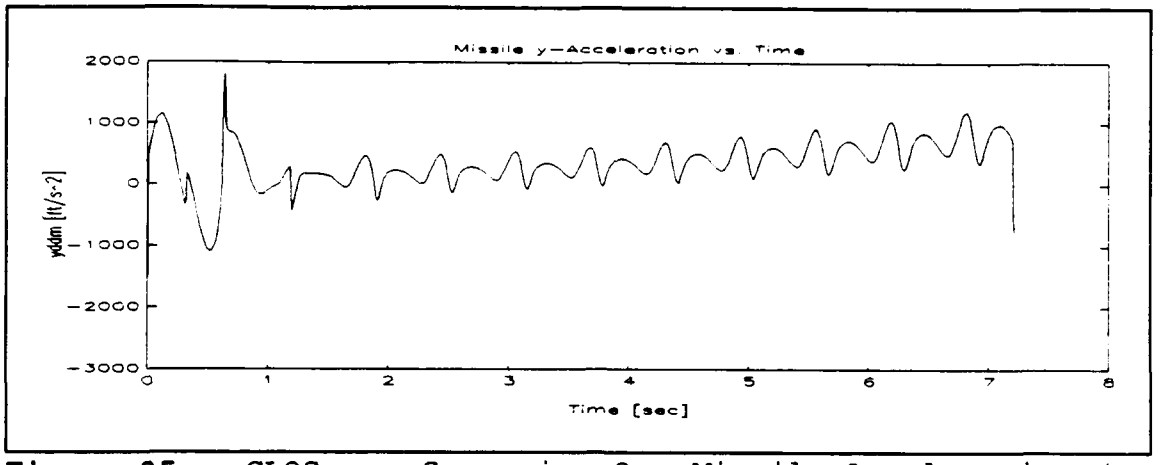


Figure 95. CLOS Scenario 2: Missile Acceleration (y-component) Time History

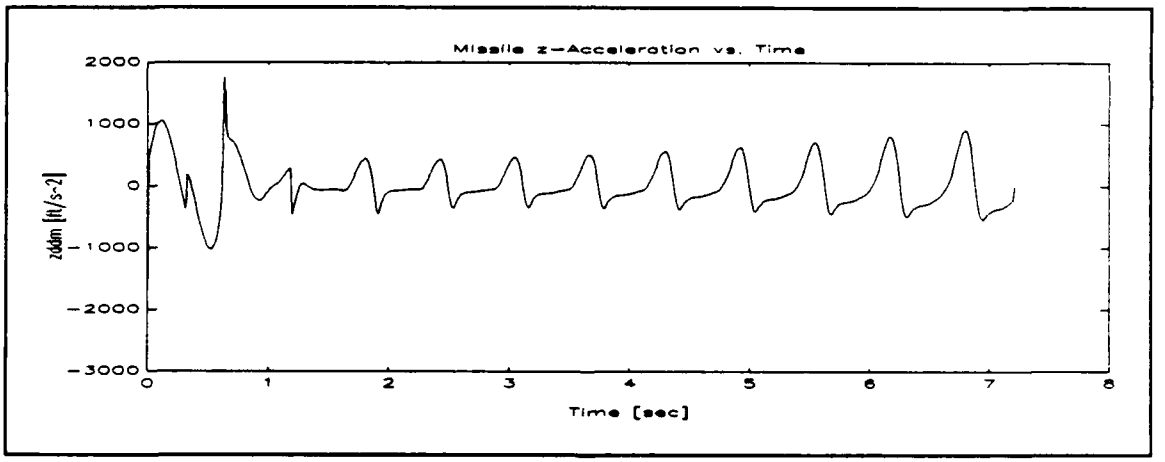


Figure 96. CLOS Scenario 2: Missile Acceleration (z-component) Time History

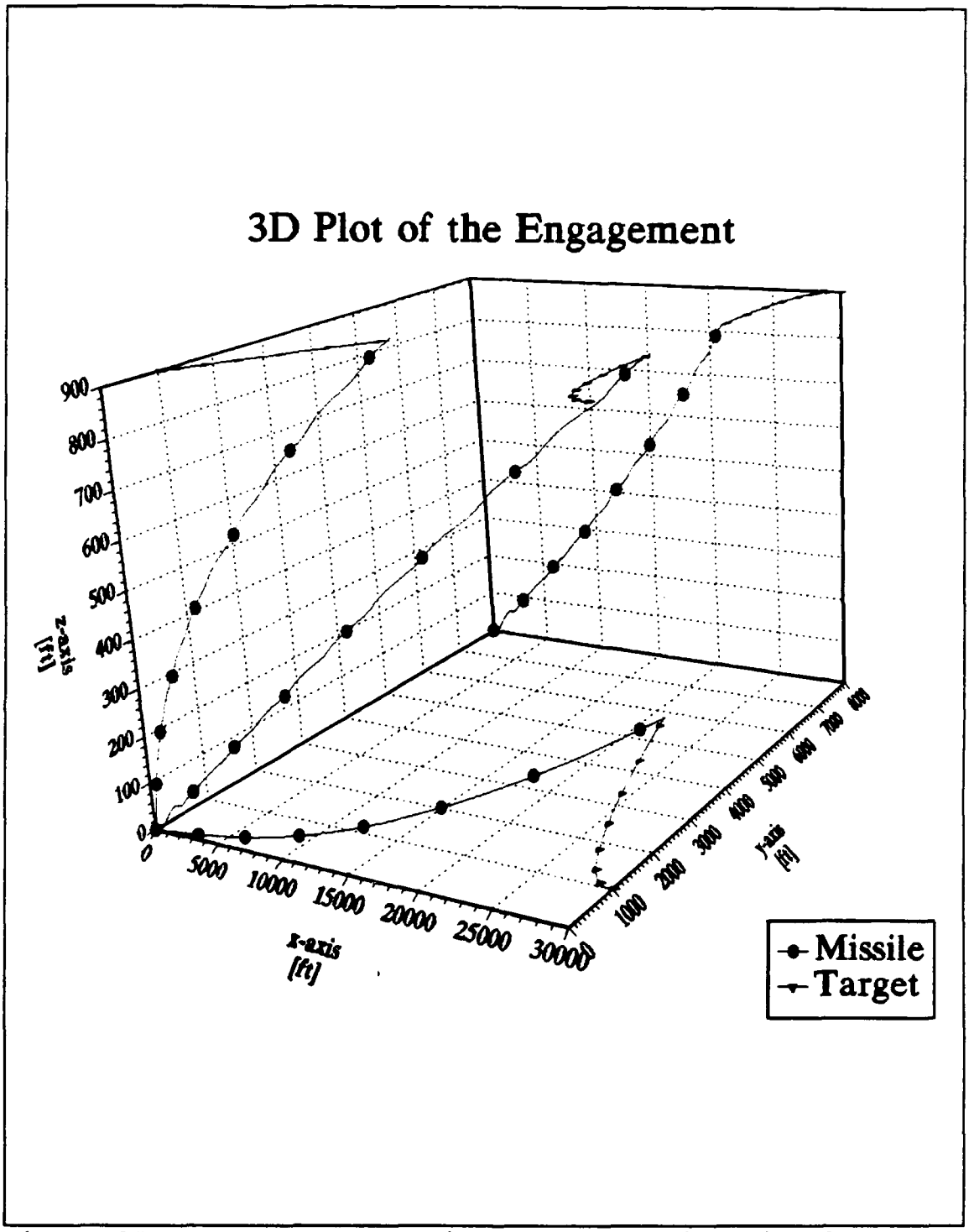


Figure 97. CLOS Scenario 2: Three-Dimensional Plot of the Engagement

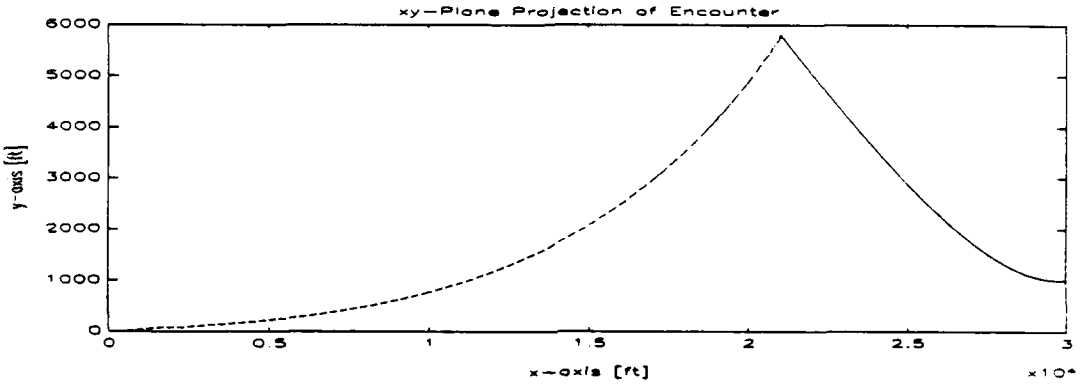


Figure 98. CLOS Scenario 2: Yaw Plane Projection of the Engagement

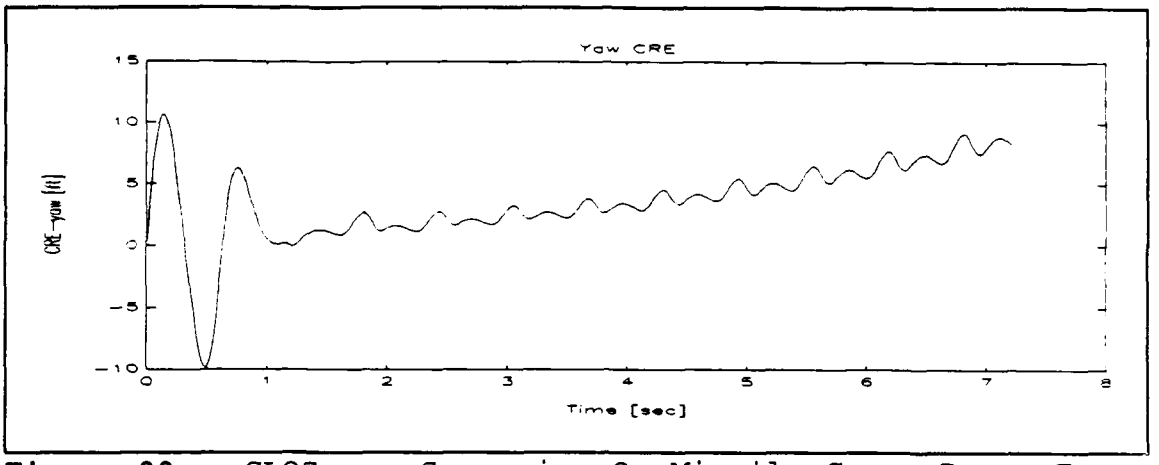


Figure 99. CLOS Scenario 2: Missile Cross Range Error (Yaw component) Time History

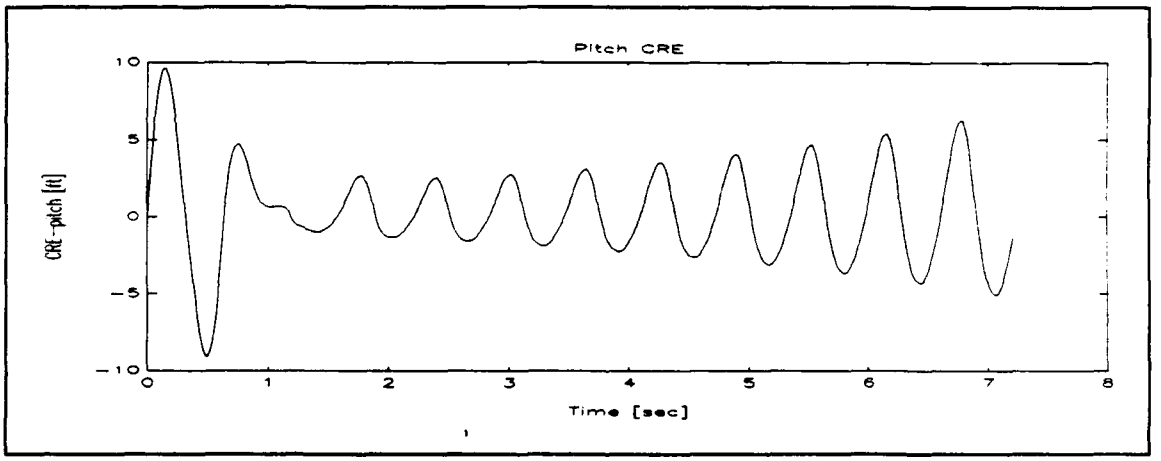


Figure 100. CLOS Scenario 2: Missile Cross Range Error (Pitch component) Time History

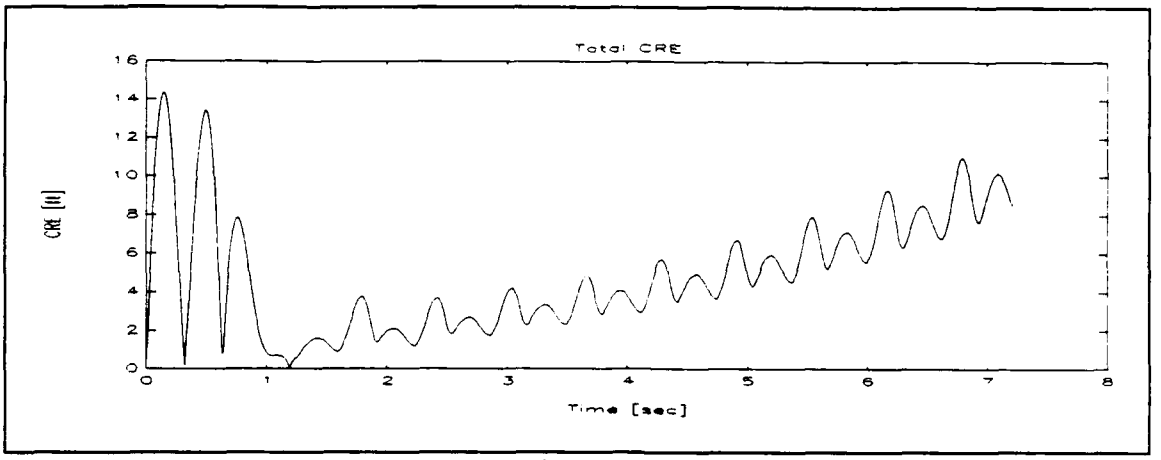


Figure 101. CLOS Scenario 2: Total Missile Cross Range Error Time History

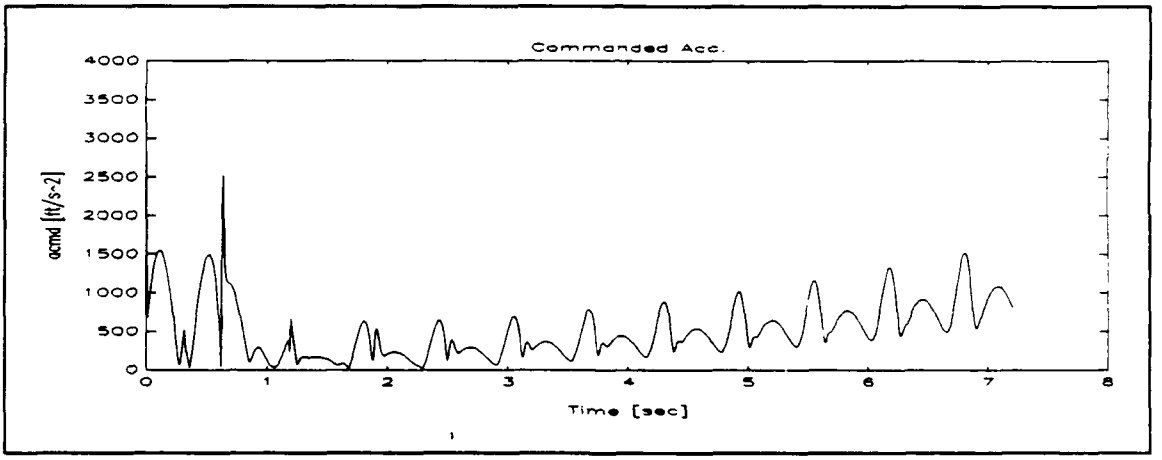


Figure 102. CLOS Scenario 2: Total Commanded Missile Acceleration Time History

A. COMPARISONS

1. Scenario 1

For the case of the level flying target the proportional navigation algorithm gave (after a few computer runs) the following results:

- Miss Distance 31.7 [ft]
- t_{CPA} 7.52 [sec]

The CLOS algorithm gave:

- Miss Distance 19.3 [ft]
- t_{CPA} 7.57 [sec]

The difference in miss distance can be found in the missile acceleration. The total missile acceleration plots show the absolute values. Whereas the proportional navigation guidance commands large values in the first second of flight, these values are much smaller than those commanded by the CLOS. In the case of the CLOS, these commands are issued each time the missile is off-beam. Also, when the illuminator-to-target and illuminator-to-missile ranges are of the same order, range resolution requires more off-beam (CRE) error. Then the commanded acceleration values are even higher than before. The total CRE, at the time of closest point of approach, is then the miss distance.

There is no distinct difference in the total flight time, but all computer runs showed a consistent lower flight time for the proportional navigation guidance in the order of hundredth of a second.

2. Scenario 2

For the case of the maneuvering target the proportional navigation algorithm gave (after a few computer runs) the following results:

- Miss Distance 63.4 [ft]
- t_{CPA} 7.20 [sec]

The CLOS algorithm gave:

- Miss Distance 12.1 [ft]
- t_{CPA} 7.21 [sec]

The same miss distance analysis, outlined in the previous subsection, holds here also. But in this case, the proportional navigation guidance gives a much higher miss than that of the CLOS.

The large acceleration demands impose large pitch and yaw angles. These, in turn, modify the velocity vectors, and finally the trajectory.

V. CONCLUSIONS AND RECOMMENDATIONS

A. CONCLUSIONS

Although the proportional navigation scheme may be an excellent tool for inertial guided missiles, command guidance offers attractive alternates.

In the case of beam riding, the missile does not require a seeker, thus reducing the unit production cost. On the other hand, the illuminator is required to remain occupied with the target until intercept. Also the accuracy varies inversely proportional with the range of intercept;i.e. targets engaged close to the illuminator produce higher range resolution than targets further away.

The algorithms developed provide an insight to the two different guidances. Proportional navigation gave higher miss distances as the target increased its maneuverability, whereas CLOS remained consistent throughout.

B. RECOMMENDATIONS

The application of an acceleration limit, would be an easy addition to the algorithms and would provide further insight to the comparison.

The implementation of noise in the measured angles, would further approximate real world conditions.

A Monte Carlo statistical analysis, would provide a further insight to the accuracy of the results.

Finally, an adjoint model analysis, would provide an excellent result validation tool.

APPENDIX A - PROPORTIONAL NAVIGATION CODE

```
%Title          :  Three (3) dimensional Command
%                  Guidance (PN)
%Author         :  Dimitrios Peppas
%Date           :  10/01/92
%Revised        :  12/01/92
%Project        :  Thesis
%System         :  Legend 386SX
%Compiler       :  386 - MATLAB v.: 3.5m
%Description    :  This code solves a 3D target/missile
%                  problem using a proportional navigation
%                  command guided missile (Forward Time
%                  Model).

clg
clear
!del cg*.met;
!del cg3dm.dat;
!del cg3dt.dat;
clc
runtime=clock; %Initialization for calculation of runtime
rtod=180/pi;   %Rad-to-Degree Conversion factor
flops(0);      %Reset Floating Point Operations Counter
```

```
%STATE DEFINITIONS
%M i s s i l e
%ms=[xm     missile x coordinate            [ft]
%    xdm    missile x velocity              [ft/s]
%    ym     missile y coordinate            [ft]
%    ydm    missile y velocity              [ft/s]
%    zm     missile z coordinate            [ft]
%    zdm]   missile z velocity              [ft/s]
%um=[xddm   missile x acceleration          [ft/s^2]
%    yddm   missile y acceleration          [ft/s^2]
%    zddm]  missile z acceleration          [ft/s^2]
%Recall :                (ms)dot=AM*(ms)+BM*um
AM=[0 1 0 0 0 0
    0 0 0 0 0 0
    0 0 0 1 0 0
    0 0 0 0 0 0
    0 0 0 0 0 1
    0 0 0 0 0 0];
BM=[0 0 0
    1 0 0
    0 0 0
    0 1 0
    0 0 0
    0 0 1];
```

```
%T a r g e t
%ts=[xt     target x coordinate              [ft]
%    xdt    target x velocity                [ft/s]
%    yt     target y coordinate              [ft]
%    ydt    target y velocity                [ft/s]
%    zt     target z coordinate              [ft]
%    zdt]   target z velocity                [ft/s]
%ut=[xddt   target x acceleration            [ft/s^2]
%    yddt   target y acceleration            [ft/s^2]
%    zddt]  target z acceleration            [ft/s^2]
%Recall :                (ts)dot=AT*(ts)+BT*ut
AT=AM;
BT=BM;
%T i m e   a n d   N a v i g a t i o n   C o n s t a n t s
%ttrack    F.C.S. tracking constant         [sec]
%tmc       F.C.S. missile command constant  [sec]
%N         F.C.S. Navigation constant       [dimensionless]
ttrack=0.1;
tmc=1;
N=4;
k1=(1/ttrack)^2;
k2=(2/ttrack);
k=(1/tmc);
%F.C.S.   T r a c k e r   a n d   F i l t e r
```

```
%trks=[beta_pitch   Tracker pitch angle estim.       [rad]
%      betad_pitch  Tracker pitch rate estim.       [rad/s]
%      beta_yaw     Tracker yaw angle estim.        [rad]
%      betad_yaw]   Tracker yaw rate estim.         [rad/s]
%utr=[sigma_pitch   Missile-to-Target pitch angle   [rad]
%     sigma_yaw]    Missile-to-Target yaw angle     [rad]
ATR=[ 0    1   0    0
     -k1  -k2  0    0
      0    0   0    1
      0    0  -k1  -k2];
BTR=[0    0
     k1   0
     0    0
     0   k1];
%Recall :                (trks)dot=ATR*trks+BTR*utr
%                  This is Equation (3.12)
%F.C.S.   G u i d a n c e  a n d  A u t o p i l o t
%mc=[gammad_pitch   Missile pitch rate command      [rad/s]
%    gammad_yaw]    Missile yaw rate command       [rad/s]
%umc=[betad_pitch
%     betad_yaw]
AMC=[-k   0
     0   -k];
BMC=[N 0
```

```
     0 N];
%Recall :                (mc)dot=AMC*mc+BMC*umc
%                   This is Equation (3.30)
%D i s c r e t i z a t i o n
dt=.01;              %Integration Interval          [sec]
[phim,delm]=c2d(AM,BM,dt);
[phit,delt]=c2d(AT,BT,dt);
[phitr,deltr]=c2d(ATR,BTR,dt);
[phimc,delmc]=c2d(AMC,BMC,dt);
tfinal=15.0;         %Calculation time range          [sec]
kmax=tfinal/dt+1;   %Maximum main loop runs
%I n i t i a l i z a t i o n
ms(:,1)=[ 0
          3000
           0
           0
           0
           0  ];
ts(:,1)=[30000
          -999.445
          1000
          -33.315
          500
          0       ];
```

```
trks(:,1)=[0
           0
           0
           0];
mc(:,1)=[0
         0];
Rt(1)=sqrt(ts(1,1)^2+ts(3,1)^2+ts(5,1)^2);
Rm(1)=sqrt(ms(1,1)^2+ms(3,1)^2+ms(5,1)^2);
R(1)=sqrt((ms(1,1)-ts(1,1))^2+(ms(3,1)-ts(3,1))^2+...
     (ms(5,1)-ts(5,1))^2);
time(1)=0;
%S i m u l a t i o n
for (i=1:kmax-1)
%                          Missile and Target Velocities
    vt(i)=sqrt(ts(2,i)^2+ts(4,i)^2+ts(6,i)^2);
    vm(i)=sqrt(ms(2,i)^2+ms(4,i)^2+ms(6,i)^2);
%                          Lines-of-Sight
    sigmam_pitch(i)=atan2(ms(5,i),sqrt(ms(1,i)^2+...
                    ms(3,i)^2));
    sigmat_pitch(i)=atan2(ts(5,i),sqrt(ts(1,i)^2+...
                    ts(3,i)^2));
    sigmam_yaw(i)=atan2(ms(3,i),ms(1,i));
    sigmat_yaw(i)=atan2(ts(3,i),ts(1,i));
    sigma_pitch(i)=atan2((ts(5,i)-ms(5,i)),...
```

```
                    sqrt((ts(1,i)^2-ms(1,i)^2)+...
                    (ts(3,i)^2-ms(3,i)^2)));
    sigma_yaw(i)=atan2((ts(3,i)-ms(3,i)),(ts(1,i)-ms(1,i)));
    utr=[sigma_pitch(i)
         sigma_yaw(i)  ];
%                        F.C.S. Tracker update
    trks(:,i+1)=phitr*trks(:,i)+deltr*utr;
%                       F.C.S. Missile Command Control update
    umc=[trks(2,i)
         trks(4,i)];
%                       Missile and Target Flight Path angles
    gammam_pitch(i)=atan2(ms(6,i),...
                    sqrt(ms(2,i)^2+ms(4,i)^2));
    gammat_pitch(i)=atan2(ts(6,i),...
                    sqrt(ts(2,i)^2+ts(4,i)^2));
    gammam_yaw(i)=atan2(ms(4,i),ms(2,i));
    gammat_yaw(i)=atan2(ts(4,i),ts(2,i));
%                        F.C.S. Missile Command update
    mc(:,i+1)=phimc*mc(:,i)+delmc*umc;
%          Missile Velocity and Acceleration in Pitch Plane
    vm_pitch(i)=vm(i)*cos(gammam_yaw(i)-sigmam_yaw(i));
    am_pitch(i)=vm_pitch(i)*mc(1,i);
%                       Missile Pitch Acceleration components
    xddm_pitch(i)=-am_pitch(i)*sin(sigma_pitch(i))*...
```

```
                cos(sigma_yaw(i));
    yddm_pitch(i)=-am_pitch(i)*sin(sigma_pitch(i))*...
                sin(sigma_yaw(i));
    zddm_pitch(i)=am_pitch(i)*cos(sigma_pitch(i));
%           Missile Velocity and Acceleration in Yaw Plane
    vm_yaw(i)=vm(i)*cos(gammam_pitch(i));
    am_yaw(i)=vm_yaw(i)*mc(2,i);
%                       Missile Yaw Acceleration components
    xddm_yaw(i)=-am_yaw(i)*sin(sigma_yaw(i));
    yddm_yaw(i)=am_yaw(i)*cos(sigma_yaw(i));
%                       Missile Acceleration components
    xddm(i)=xddm_pitch(i)+xddm_yaw(i);
    yddm(i)=yddm_pitch(i)+yddm_yaw(i);
    zddm(i)=zddm_pitch(i);
    um=[xddm(i)
        yddm(i)
        zddm(i)];
    am(i)=sqrt(um(1)^2+um(2)^2+um(3)^2);
%                       Missile update
    ms(:,i+1)=phim*ms(:,i)+delm*um;
%                       Target Acceleration components
    xddt(i)=0;%-6.5*32.2*sin(gammat_yaw(i));
    yddt(i)=0;%-6.5*32.2*cos(gammat_yaw(i));
    zddt(i)=0;%-0.1*32.2*cos(gammat_pitch(i));
```

```
    ut=[xddt(i)
        yddt(i)
        zddt(i)];
    at(i)=sqrt(ut(1)^2+ut(2)^2+ut(3)^2);
%                       Target update
    ts(:,i+1)=phit*ts(:,i)+delt*ut;
%                       Time-to-go
    vt_yaw(i)=vt(i)*cos(gammat_pitch(i));
    vc(i)=-(vt_yaw(i)*cos(gammat_yaw(i)-sigma_yaw(i))...
          -vm_yaw(i)*cos(gammam_yaw(i)-sigma_yaw(i)));
    ttg(i)=R(i)/vc(i);
%                       Range update
    Rt(i+1)=sqrt(ts(1,i+1)^2+ts(3,i+1)^2+ts(5,i+1)^2);
    Rm(i+1)=sqrt(ms(1,i+1)^2+ms(3,i+1)^2+ms(5,i+1)^2);
    R(i+1)=sqrt((ms(1,i+1)-ts(1,i+1))^2+...
           (ms(3,i+1)-ts(3,i+1))^2+...
           (ms(5,i+1)-ts(5,i+1))^2);
%                       Time update
    time(i+1)=time(i)+dt;
%                       Rmiss check
    if (R(i) < R(i+1))
        tcpa=time(i);   %Time-of-C.P.A.
        break
    end
```

```
end
flop_count=flops;
runtime=etime(clock,runtime)
%F i l i n g
X=[ms(1,:)' ms(3,:)' ms(5,:)'];
Y=[ts(1,:)' ts(3,:)' ts(5,:)'];
save cg3dm.dat X /ascii ;
save cg3dt.dat Y /ascii ;
%P r i n t o u t
plot(ttg(1:i-1),R(1:i-1));
title('Miss Distance vs. Time-to-go');
xlabel('TTG [sec]');ylabel('Rmiss [ft]');
meta cg1
pause

clg
plot(time,ts(1,:),time,ms(1,:));
title('x-coordinate Time History');
xlabel('Time [sec]');ylabel('x-coord. [ft]');
gtext('Missile');gtext('Target');
meta cg2
pause

clg
```

```
plot(time,ts(3,:),time,ms(3,:));
title('y-coordinate Time History');
xlabel('Time [sec]');ylabel('y-coord. [ft]');
gtext('Missile');gtext('Target');
meta cg3
pause

clg
plot(time,ts(5,:),time,ms(5,:));
title('z-coordinate Time History');
xlabel('Time [sec]');ylabel('z-coord. [ft]');
gtext('Missile');gtext('Target');
meta cg4
pause

clg
plot(time,R);
title('Miss Distance vs. Time');
xlabel('Time [sec]');ylabel('Rmiss [ft]');
meta cg5
pause

clg
plot(time,ts(2,:),time,ms(2,:));
```

```
title('x-Velocity Time History');
xlabel('Time [sec]');ylabel('Vx [ft/s]');
gtext('Missile');gtext('Target');
meta cg6
pause

clg
plot(time,ts(4,:),time,ms(4,:));
title('y-Velocity Time History');
xlabel('Time [sec]');ylabel('Vy [ft/s]');
gtext('Missile');gtext('Target');
meta cg7
pause

clg
plot(time,ts(6,:),time,ms(6,:));
title('z-Velocity Time History');
xlabel('Time [sec]');ylabel('Vz [ft/s]');
gtext('Missile');gtext('Target');
meta cg8
pause

clg
plot(time(1:i),rtod*gammam_pitch);
```

```
title('Missile Pitch Angle vs. Time');
xlabel('Time [sec]');ylabel('gammam_pitch [deg]');
meta cg9
pause

clg
plot(time(1:i),rtod*gammam_yaw);
title('Missile Yaw Angle vs. Time');
xlabel('Time [sec]');ylabel('gammam_yaw [deg]');
meta cg10
pause

clg
plot(time,rtod*mc(1,:));
title('Missile Pitch Rate vs. Time');
xlabel('Time [sec]');ylabel('gammamd_pitch [deg/s]');
meta cg11
pause

clg
plot(time,rtod*mc(2,:));
title('Missile Yaw Rate vs. Time');
xlabel('Time [sec]');ylabel('gammamd_yaw [deg/s]');
meta cg12
```

```
pause

%Following plots were not included in the documentation
%clg
%plot(time(1:i),rtod*gammat_pitch);
%title('Target Pitch Angle vs. Time');
%xlabel('Time [sec]');ylabel('gammat_pitch [deg]');
%meta cg13
%pause
%clg
%plot(time(1:i),rtod*gammat_yaw);
%title('Target Yaw Angle vs. Time');
%xlabel('Time [sec]');ylabel('gammat_yaw [deg]');
%meta cg14
%pause

clg
plot(time(1:i),vc);
title('Closing Velocity vs. Time');
xlabel('Time [sec]');ylabel('Vc [ft/s]');
meta cg15
pause

clg
```

```
axis([tcpa-.1 tcpa+.1 0 min(R)+20]);
plot(time,R);
title('Rmiss "Zoom" Plot');axis;
xlabel('Time [sec]');ylabel('Rmiss [ft]');
meta cg16
pause

clg
plot(time(1:i),am);
title('Total Missile Acceleration vs. Time');
xlabel('Time [sec]');ylabel('am [ft/s^2]');
meta cg17
pause

clg
plot(time(1:i),xddm);
title('Missile x-Acceleration vs. Time');
xlabel('Time [sec]');ylabel('xddm [ft/s^2]');
meta cg18
pause

clg
plot(time(1:i),yddm);
title('Missile y-Acceleration vs. Time');
```

```
xlabel('Time [sec]');ylabel('yddm [ft/s^2]');
meta cg19
pause

clg
plot(time(1:i),zddm);
title('Missile z-Acceleration vs. Time');
xlabel('Time [sec]');ylabel('zddm [ft/s^2]');
meta cg20
pause

%Following plots were not included in the documentation
%clg
%plot(time(1:i),at);
%title('Total Target Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('at [ft/s^2]');
%meta cg21
%pause
%clg
%plot(time(1:i),xddt);
%title('Target x-Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('xddt [ft/s^2]');
%meta cg22
%pause
```

```
%clg
%plot(time(1:i),yddt);
%title('Target y-Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('yddt [ft/s^2]');
%meta cg23
%pause
%clg
%plot(time(1:i),zddt);
%title('Target z-Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('zddt [ft/s^2]');
%meta cg24
%pause

clg
plot(ts(1,:),ts(3,:),'r',ms(1,:),ms(3,:),'g');
title('xy-Plane Projection of Encounter');
xlabel('x-axis [ft]');ylabel('y-axis [ft]');
meta cg26
```

APPENDIX B - CLOS CODE

```
%Title          :  Three (3)D Command Guidance (CLOS)
%Author         :  Dimitrios Peppas
%Date           :  10/01/92
%Revised        :  11/30/92
%Project        :  Thesis
%System         :  Legend 386SX
%Compiler       :  386 - MATLAB v.: 3.5m
%Description    :  This code solves a  3D  target/missile
%                  problem using a command-to-line-of-sight
%                  command guided missile (Forward Time
%                  Model).

clg
clear
!del clg*.met;
!del cl3dm.dat;
!del cl3dt.dat;
clc
runtime=clock;     %Initialization for calculation of runtime
rtod=180/pi;       %Rad-to-Degree Conversion factor
flops(0);          %Reset Floating Point Operations Counter
```

```
%STATE DEFINITIONS
%M i s s i l e
%ms=[xm            missile x coordinate          [ft]
%    xdm           missile x velocity            [ft/s]
%    ym            missile y coordinate          [ft]
%    ydm           missile y velocity            [ft/s]
%    zm            missile z coordinate          [ft]
%    zdm]          missile z velocity            [ft/s]
%um=[xddm          missile x acceleration        [ft/s^2]
%    yddm          missile y acceleration        [ft/s^2]
%    zddm]         missile z acceleration        [ft/s^2]
%Recall :                (ms)dot=AM*(ms)+BM*um
AM=[0 1 0 0 0 0
    0 0 0 0 0 0
    0 0 0 1 0 0
    0 0 0 0 0 0
    0 0 0 0 0 1
    0 0 0 0 0 0];
BM=[0 0 0
    1 0 0
    0 0 0
    0 1 0
    0 0 0
    0 0 1];
```

```
%T a r g e t
%ts=[xt          target x coordinate          [ft]
%    xdt         target x velocity            [ft/s]
%    yt          target y coordinate          [ft]
%    ydt         target y velocity            [ft/s]
%    zt          target z coordinate          [ft]
%    zdt]        target z velocity            [ft/s]
%ut=[xddt        target x acceleration        [ft/s^2]
%    yddt        target y acceleration        [ft/s^2]
%    zddt]       target z acceleration        [ft/s^2]
%Recall :              (ts)dot=AT*(ts)+BT*ut
AT=AM;
BT=BM;
%D i s c r e t i z a t i o n
dt=.01;          %Intergration Interval       [sec]
[phim,delm]=c2d(AM,BM,dt);
[phit,delt]=c2d(AT,BT,dt);
tfinal=15.0;     %Calculation time range      [sec]
kmax=tfinal/dt+1; %Maximum main loop runs
%I n i t i a l i z a t i o n
ms(:,1)=[ eps
         3000
          eps
           0
```

```
            0
            0  ];
ts(:,1)=[30000
         -999.445
          1000
         -33.315
          900
            0  ];
Rt(1)=sqrt(ts(1,1)^2+ts(3,1)^2+ts(5,1)^2);
Rm(1)=sqrt(ms(1,1)^2+ms(3,1)^2+ms(5,1)^2);
R(1)=sqrt((ms(1,1)-ts(1,1))^2+(ms(3,1)-ts(3,1))^2+(ms(5,1)-t
s(5,1))^2);
time(1)=0;
%S i m u l a t i o n
for (i=1:kmax-1)
%                          Missile and Target Velocities
    vt(i)=sqrt(ts(2,i)^2+ts(4,i)^2+ts(6,i)^2);
    vm(i)=sqrt(ms(2,i)^2+ms(4,i)^2+ms(6,i)^2);
%                          Lines-of-Sight
    sigmam_pitch(i)=atan2(ms(5,i),...
                          sqrt(ms(1,i)^2+ms(3,i)^2));
    sigmat_pitch(i)=atan2(ts(5,i),...
                          sqrt(ts(1,i)^2+ts(3,i)^2));
    sigmam_yaw(i)=atan2(ms(3,i),ms(1,i));
```

```
sigmat_yaw(i)=atan2(ts(3,i),ts(1,i));
sigma_pitch(i)=atan2((ts(5,i)-ms(5,i)),...
                  sqrt((ts(1,i)^2-ms(1,i)^2)+...
                 (ts(3,i)^2-ms(3,i)^2)));
sigma_yaw(i)=atan2((ts(3,i)-ms(3,i)),(ts(1,i)-ms(1,i)));
%                          C.R.E. and (C.R.E.)'
g1a=(ms(3,i)*ts(5,i)-ms(5,i)*ts(3,i));
g1b=-(ms(5,i)*ts(1,i)-ms(1,i)*ts(5,i));
g1c=(ms(1,i)*ts(3,i)-ms(3,i)*ts(1,i));
g1=g1a^2+g1b^2+g1c^2;
g2a=2*g1a*(ms(4,i)*ts(5,i)+ms(3,i)*ts(6,i)-...
     ms(6,i)*ts(3,i)-ms(5,i)*ts(4,i));
g2b=2*g1b*(ms(6,i)*ts(1,i)+ms(5,i)*ts(2,i)-...
    ms(2,i)*ts(5,i)-ms(1,i)*ts(6,i));
g2c=2*g1c*(ms(2,i)*ts(3,i)+ms(1,i)*ts(4,i)-...
    ms(4,i)*ts(1,i)-ms(3,i)*ts(2,i));
g2=g2a+g2b+g2c;
f2=(ms(1,i)*ms(2,i)+ms(3,i)*ms(4,i)+...
   ms(5,i)*ms(6,i))/Rt(i);
CRE(i)=sqrt(g1)/Rt(i);
CRED(i)=((1/2)*(g2/sqrt(g1))*Rt(i)-...
         f2*sqrt(g1))/(Rt(i)^2);
CRE_yaw(i)=sqrt(ms(1,i)^2+ms(3,i)^2)*...
             sin(sigmat_yaw(i)-sigmam_yaw(i));
```

```
CRE_pitch(i)=sqrt(CRE(i)^2-CRE_yaw(i)^2)*...
                sign(sigmat_pitch(i)-sigmam_pitch(i));
%                       Missile and Target Flight Path angles
gammam_pitch(i)=atan2(ms(6,i),...
                      sqrt(ms(2,i)^2+ms(4,i)^2));
gammat_pitch(i)=atan2(ts(6,i),...
                      sqrt(ts(2,i)^2+ts(4,i)^2));
gammam_yaw(i)=atan2(ms(4,i),ms(2,i));
gammat_yaw(i)=atan2(ts(4,i),ts(2,i));
%                       Acceleration Command
acmd_yaw(i)=-100*CRE_yaw(i)-25*CRED(i);
acmd_pitch(i)=-100*CRE_pitch(i)-25*CRED(i);
%          Missile Velocity and Acceleration in Pitch Plane
vm_pitch(i)=(vm(i)*cos(gammam_yaw(i)-sigmam_yaw(i)));
am_pitch(i)=-acmd_pitch(i);
%                       Missile Pitch Acceleration components
xddm_pitch(i)=-am_pitch(i)*sin(sigma_pitch(i))*...
                cos(sigma_yaw(i));
yddm_pitch(i)=-am_pitch(i)*sin(sigma_pitch(i))*...
                sin(sigma_yaw(i));
zddm_pitch(i)=am_pitch(i)*cos(sigma_pitch(i));
%             Missile Velocity and Acceleration in Yaw Plane
vm_yaw(i)=vm(i)*cos(gammam_pitch(i));
am_yaw(i)=-acmd_yaw(i);
```

```
%                          Missile Yaw Acceleration components
    xddm_yaw(i)=-am_yaw(i)*sin(sigma_yaw(i));
    yddm_yaw(i)=am_yaw(i)*cos(sigma_yaw(i));
%                          Missile Acceleration components
    xddm(i)=xddm_pitch(i)+xddm_yaw(i);
    yddm(i)=yddm_pitch(i)+yddm_yaw(i);
    zddm(i)=zddm_pitch(i);
    um=[xddm(i)
        yddm(i)
        zddm(i)];
    am(i)=sqrt(um(1)^2+um(2)^2+um(3)^2);
%                          Missile update
    ms(:,i+1)=phim*ms(:,i)+delm*um;
%                          Target Acceleration components
    xddt(i)=-6.5*32.2*sin(gammat_yaw(i));%0;
    yddt(i)=-6.5*32.2*cos(gammat_yaw(i));%0;
    zddt(i)=-0.1*32.2*cos(gammat_pitch(i));%0;
    ut=[xddt(i)
        yddt(i)
        zddt(i)];
    at(i)=sqrt(ut(1)^2+ut(2)^2+ut(3)^2);
%                          Target update
    ts(:,i+1)=phit*ts(:,i)+delt*ut;
%                          Time-to-go
```

```
    vt_yaw(i)=vt(i)*cos(gammat_pitch(i));
    vc(i)=-(vt_yaw(i)*cos(gammat_yaw(i)-sigma_yaw(i))-...
           vm_yaw(i)*cos(gammam_yaw(i)-sigma_yaw(i)));
    ttg(i)=abs(R(i)/vc(i));
%                          Range update
    Rt(i+1)=sqrt(ts(1,i+1)^2+ts(3,i+1)^2+ts(5,i+1)^2);
    Rm(i+1)=sqrt(ms(1,i+1)^2+ms(3,i+1)^2+ms(5,i+1)^2);
    R(i+1)=sqrt((ms(1,i+1)-ts(1,i+1))^2+...
           (ms(3,i+1)-ts(3,i+1))^2+...
           (ms(5,i+1)-ts(5,i+1))^2);
%                          Time update
    time(i+1)=time(i)+dt;
%                          Rmiss check
    if (R(i) < R(i+1))
        tcpa=time(i);    %Time-of-C.P.A.
        break
    end
end
flop_count=flops;
runtime=etime(clock,runtime)
%F i l i n g
X=[ms(1,:)' ms(3,:)' ms(5,:)'];
Y=[ts(1,:)' ts(3,:)' ts(5,:)'];
save cl3dm.dat X /ascii;
```

```
save c13dt.dat Y /ascii;
%P r i n t o u t
plot(ttg(1:i-1),R(1:i-1));
title('Miss Distance vs. Time-to-go');
xlabel('TTG [sec]');ylabel('Rmiss [ft]');
meta clg1
pause

clg
plot(time,ts(1,:),time,ms(1,:));
title('x-coordinate Time History');
xlabel('Time [sec]');ylabel('x-coord. [ft]');
gtext('Missile');gtext('Target');
meta clg2
pause

clg
plot(time,ts(3,:),time,ms(3,:));
title('y-coordinate Time History');
xlabel('Time [sec]');ylabel('y-coord. [ft]');
gtext('Missile');gtext('Target');
meta clg3
pause
```

```
clg
plot(time,ts(5,:),time,ms(5,:));
title('z-coordinate Time History');
xlabel('Time [sec]');ylabel('z-coord. [ft]');
gtext('Missile');gtext('Target');
meta clg4
pause

clg
plot(time,R);
title('Miss Distance vs. Time');
xlabel('Time [sec]');ylabel('Rmiss [ft]');
meta clg5
pause

clg
plot(time,ts(2,:),time,ms(2,:));
title('x-Velocity Time History');
xlabel('Time [sec]');ylabel('Vx [ft/s]');
gtext('Missile');gtext('Target');
meta clg6
pause

clg
```

```
plot(time,ts(4,:),time,ms(4,:));
title('y-Velocity Time History');
xlabel('Time [sec]');ylabel('Vy [ft/s]');
gtext('Missile');gtext('Target');
meta clg7
pause

clg
plot(time,ts(6,:),time,ms(6,:));
title('z-Velocity Time History');
xlabel('Time [sec]');ylabel('Vz [ft/s]');
gtext('Missile');gtext('Target');
meta clg8
pause

clg
plot(time(1:i),rtod*gammam_pitch);
title('Missile Pitch Angle vs. Time');
xlabel('Time [sec]');ylabel('gammam_pitch [deg]');
meta clg9
pause

clg
plot(time(1:i),rtod*gammam_yaw);
```

```
title('Missile Yaw Angle vs. Time');
xlabel('Time [sec]');ylabel('gammam_yaw [deg]');
meta clg10
pause

%%Following plots were not included in the documentation
%clg
%plot(time(1:i),rtod*gammat_pitch);
%title('Target Pitch Angle vs. Time');
%xlabel('Time [sec]');ylabel('gammat_pitch [deg]');
%meta clg13
%pause
%clg
%plot(time(1:i),rtod*gammat_yaw);
%title('Target Yaw Angle vs. Time');
%xlabel('Time [sec]');ylabel('gammat_yaw [deg]');
%meta clg14
%pause

clg
plot(time(1:i),vc);
title('Closing Velocity vs. Time');
xlabel('Time [sec]');ylabel('Vc [ft/s]');
meta clg15
```

```
pause

clg
axis([tcpa-.1 tcpa+.1 0 min(R)+20]);
plot(time,R);
title('Rmiss "Zoom" Plot');axis;
xlabel('Time [sec]');ylabel('Rmiss [ft]');
meta clg16
pause

clg
plot(time(1:i),am);
title('Total Missile Acceleration vs. Time');
xlabel('Time [sec]');ylabel('am [ft/s^2]');
meta clg17
pause

clg
plot(time(1:i),xddm);
title('Missile x-Acceleration vs. Time');
xlabel('Time [sec]');ylabel('xddm [ft/s^2]');
meta clg18
pause
```

```
clg
plot(time(1:i),yddm);
title('Missile y-Acceleration vs. Time');
xlabel('Time [sec]');ylabel('yddm [ft/s^2]');
meta clg19
pause

clg
plot(time(1:i),zddm);
title('Missile z-Acceleration vs. Time');
xlabel('Time [sec]');ylabel('zddm [ft/s^2]');
meta clg20
pause

%Following plots were not included in the documentation
%clg
%plot(time(1:i),at);
%title('Total Target Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('at [ft/s^2]');
%meta clg21
%pause
%clg
%plot(time(1:i),xddt);
%title('Target x-Acceleration vs. Time');
```

```
%xlabel('Time [sec]');ylabel('xddt [ft/s^2]');
%meta clg22
%pause
%clg
%plot(time(1:i),yddt);
%title('Target y-Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('yddt [ft/s^2]');
%meta clg23
%pause
%clg
%plot(time(1:i),zddt);
%title('Target z-Acceleration vs. Time');
%xlabel('Time [sec]');ylabel('zddt [ft/s^2]');
%meta clg24
%pause

clg
plot(ts(1,:),ts(3,:),'r',ms(1,:),ms(3,:),'g');
title('xy-Plane Projection of Encounter');
xlabel('x-axis [ft]');ylabel('y-axis [ft]');
meta clg26
pause

clg
```

```
plot(time(1:i),CRE_yaw);
title('Yaw CRE');
xlabel('Time [sec]');ylabel('CRE_yaw [ft]');
meta clg27
pause

clg
plot(time(1:i),CRE_pitch);
title('Pitch CRE');
xlabel('Time [sec]');ylabel('CRE_pitch [ft]');
meta clg28
pause

clg
plot(time(1:i),CRE);
title('Total CRE');
xlabel('Time [sec]');ylabel('CRE [ft]');
meta clg29
pause

clg
acmd=sqrt(acmd_yaw.^2+acmd_pitch.^2);
plot(time(1:i),acmd);
title('Commanded Acc.');
```

```
xlabel('Time [sec]');ylabel('acmd [ft/s^2]');
meta clg30
```

LIST OF REFERENCES

1. Lindsey,G.H., and Redmon, D.R., *Tactical Missile Design*, unpublished notes, Naval Postgraduate School.
2. Titus, H.A., *Missile Guidance*, unpublished notes, Naval Postgraduate School.
3. Lin, C.F., *Modern Navigation, Guidance and Control Processing*, Prentice Hall, 1991.
4. Eichblatt Jr., E.J., "Test and Evaluation of the Tactical Missile, " AIAA Vol. 119, 1989.
5. Lukenbill, F.C., "A Target/Engagement Scenario Using Classical Proportional Navigation, " M.S. Thesis, Naval Postgraduate School, Monterey CA, December 1990.
6. Finney, R., and Thomas, G., *Calculus*, Addison Wesley, July 1991.
7. Yeun, J.Y., "Computer Simulated Development of a Command to Line-of-Sight Missile Using On-Off Control, " M.S. Thesis, Naval Postgraduate School, Monterey CA, December 1983.
8. Burl, J.B., *Linear Optimal Estimation and Control Notes*, Naval Postgraduate School, Monterey CA, September 1991.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Technical Information Center Cameron Station Alexandria, VA 22304-6145	2
2.	Library, Code 52 Naval Postgraduate School Monterey, CA 93943-5002	2
3.	Chairman, Code EC Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
4.	Professor H. Titus, Code EC/Ts Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
5.	Professor R. Cristi, Code EC/Cx Department of Electrical and Computer Engineering Naval Postgraduate School Monterey, CA 93943-5000	1
6.	Embassy of Greece Naval Attache 2228 Massachusetts Ave., N.W. Washington, D.C. 20008	1
7.	Lieutenant (j.g.) Dimitrios Peppas H.N. 18-22 N. Zerva Str., 15772 Athens, Greece	7